МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ
ДОНЕЦКОЙ НАРОДНОЙ РЕСПУБЛИКИ
ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
«ДОНЕЦКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

к выполнению курсового проекта
по дисциплине
«Объектно-ориентированное программирование»

Донецк
2022

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ
ДОНЕЦКОЙ НАРОДНОЙ РЕСПУБЛИКИ
ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
«ДОНЕЦКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

КАФЕДРА «КОМПЬЮТЕРНОЕ МОДЕЛИРОВАНИЕ И ДИЗАЙН»

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

к выполнению курсового проекта
по дисциплине
«Объектно-ориентированное программирование»

для обучающихся по направлениям подготовки
02.03.01 «Математика и компьютерные науки»
09.03.02 «Информационные системы и технологии»
всех форм обучения

РАССМОТРЕНО
на заседании кафедры
компьютерного моделирования и дизайна
Протокол №1 от 31.08.2022 г.

УТВЕРЖДЕНО
на заседании учебно-издательского
совета ДОННТУ
Протокол № 6 от 21.09.2022 г.

Донецк
2022

УДК 004.43:004.8(076)
М54

Составитель:
Павлий Виталий Александрович – кандидат технических наук, доцент кафедры компьютерного моделирования и дизайна ГОУВПО «ДОННТУ».

М54 **Методические указания к выполнению курсового проекта по дисциплине «Объектно-ориентированное программирование»** : для обучающихся по направлениям подготовки 02.03.01 «Математика и компьютерные науки», 09.03.02 «Информационные системы и технологии» всех форм обучения / ГОУВПО «ДОННТУ», Каф. компьютерного моделирования и дизайна ; сост. В. А. Павлий. – Донецк : ДОННТУ, 2022. – Систем. требования: Acrobat Reader. – Загл. с титул. экрана.

Методические указания разработаны с целью оказания помощи обучающимся в выполнении курсового проекта по дисциплине «Объектно-ориентированное программирование» и содержат материалы, отражающие цели и задачи выполнения курсового проекта, основные требования к разработке и оформлению пояснительной записки, а также описание процесса защиты и критерии оценивания. Методические указания также включают 50 индивидуальных вариантов заданий для выполнения курсового проекта.

УДК 004.43:004.8(076)

Содержание

Введение

Данные методические указания разработаны для оказания помощи обучающимся при выполнении курсового проекта по дисциплине «Объектно-ориентированное программирование», а также оформлении сопроводительной документации к выполняемому проекту. Они содержат подробную информацию, касающуюся требований к выполняемой работе и требований к оформлению пояснительной записки, которая является основным документом.

Выполнение курсового проекта рассчитано на стандартный учебный семестр (17 недель). При этом 14 недель отводятся на выполнение проекта, еще 2 недели – на оформление пояснительной записки. В оставшуюся неделю проходит защита курсового проекта.

Данные указания не содержат теоретических сведений (например, основ алгоритмизации, описания синтаксиса языка программирования и т.п.), необходимых для выполнения курсового проекта. Предполагается, что обучающийся уже усвоил необходимый теоретический материал в ходе выполнения лабораторных работ по соответствующим методическим указаниям и конспекту лекций, а выполнение курсового проекта направлено на практическое углубление и закрепление изученного материала. Вместе с тем основные правила оформления пояснительной записки к курсовому проекту, в упомянутых материалах обычно не рассматриваются, а потому здесь изложены максимально подробно.

1 Общие положения

Курсовой проект по дисциплине представляет собой самостоятельную законченную работу, написанную лично обучающимся под руководством преподавателя, свидетельствующую об его умении работать с литературой, обобщать и анализировать фактический материал, использовать теоретические знания и практические навыки, полученные при освоении дисциплины. Выполнение курсовых проектов предусматривается типовыми учебными планами в целях закрепления, углубления и обобщения знаний, полученных обучающимся, развития способности самостоятельного и творческого мышления.

Курсовой проект призван выявить знания обучающегося по дисциплине и умение применять эти знания в практической работе. В процессе курсового проектирования обучающийся должен проявить свои навыки к самостоятельной работе с научно-технической литературой, к обобщению накопленного опыта и своё умение делать обоснованные выводы и выдвигать рекомендации.

Курсовое проектирование является формой текущей аттестации по дисциплине учебного плана специальности. Зачет по курсовому проекту учитывается при определении общего числа экзаменов и зачетов на соответствующем этапе обучения. Темы курсовых проектов вносятся в зачетные книжки.

Основными требованиями, предъявляемыми к курсовому проекту являются:

– целевая направленность;
– четкость построения;
– логическая последовательность изложения материала;
– глубина исследования и полнота освещения материала;
– убедительность аргументаций;
– краткость и точность формулировок;
– конкретное изложение результатов работы;
– доказательность выводов и обоснованность рекомендаций.

Курсовой проект представляет собой законченную разработку, в которой сформулирована актуальность и место решаемой задачи информационного обеспечения в профессиональной области; анализируется литература и информация, полученная с помощью Интернет; определяются и конкретно описываются выбранные методы и средства решаемой задачи, иллюстрируемые данными и формами выходных документов.

От курсовой работы курсовой проект отличается в первую очередь наличием расчетно-проектной чисти. Также в курсовом проекте обязательно должна быть графическая часть, включающая один или несколько чертежей. Некоторые курсовые проекты также содержат аналитическую часть, призванную доказать, что выполненный проект эффективнее (например, быстрее работает, потребляет меньше ресурсов и т.п.) по отношению к аналогам, либо его разработка обойдется дешевле имеющихся аналогов, либо проект содержит дополнительные функции, которые не реализованы в аналогах.

2 Цели и задачи курсового проектирования

Целью выполнения курсового проекта роботы является закрепление навыков программирования при решении реальной практической задачи и углубление знаний, принципов и приемов создания GUI-приложений. При этом у обучающегося формируется полное представление об этапах проектирования и создания таких приложений, а также их последующего тестирования.

В ходе достижения цели решаются следующие задачи:

– развитие логического и алгоритмического мышления;

– закрепление знаний по терминологии, основным понятиям объектно-ориентированного программирования и элементам синтаксиса языка программирования C++;

– получение навыков проектирования объектно-ориентированных приложений, умения разбивать логику приложения на структурные единицы (библиотеки, модули, классы) и определять задачи для реализации этих структурных единиц;

– формирование умений изображать структуру проекта в виде схем, диаграмм, графиков, показывать представления объектов и связей между ними;

– закрепление навыков разработки программного кода при помощи интегрированной среды разработки, например, Visual Studio или ее аналогов;

– закрепление умений работы со сторонними библиотеками, самостоятельного изучения их возможностей по прилагаемой документации (в т.ч. на английском языке), навыков их интеграции в разрабатываемый проект;

– закрепление теоретических знаний о паттернах проектирования как типовых решений наиболее часто повторяющихся задач в разработке приложений, умений использовать их на практике;

– закрепление умений работы с встроенными средствами отладки программного кода, нахождения и исправления ошибок с их помощью;

– подготовка к освоению (возможно самостоятельному) новых возможностей объектно-ориентированного программирования, которые поддерживаются в других языках программирования, но отсутствуют в C++ (например, поддержка интерфейсов);

– подготовка к выполнению выпускной квалификационной работы.

3 Основные требования к разработке программного продукта

В ходе выполнения курсового проекта обучающийся разрабатывает **программный продукт на языке C++** с использованием объектно-ориентированного подхода. Вариант задания **может быть выбран обучающимся самостоятельно** из перечня вариантов, приводимых в данных методических указаниях, однако выбираемые обучающимися варианты **не должны повторяться** в потоке. Организацию выбора вариантов осуществляют старосты групп, которые создают для этого общую беседу в социальной сети или общий документ с фамилиями (например, Google Docs), а обучающиеся сообщают в этой беседе или вписывают напротив своей фамилии номер варианта, предварительно убедившись, что он еще никем не занят. После этого номер варианта сообщается преподавателю, ведущему практические занятия, **и обязательно лектору курса**. После этого выбранный вариант считается закрепленным за конкретным обучающимся, изменять его в дальнейшем **запрещено**. После утверждения варианта преподавателем необходимо заполнить распечатать и подписать техническое задание (см. приложение Б).

Варианты заданий условно разделены на три типа: специализированные программные продукты (ПП), моделирующие ПП и игровые ПП. В специализированных ПП требуется разработать приложение по работе с данными специального назначения, например, калькулятор многочленов или нотный редактор. В ПО моделирующего типа необходимо изобразить анимацию некоторого процесса во времени, например, загруженности перекрестка. В игровых приложениях обучающемуся предлагается разработать собственный вариант компьютерной игры. В вариантах заданий имеются описания малознакомых (но тем не менее увлекательных) игр. Классические игры (например, тетрис) также есть в вариантах заданий, однако в них имеются усложнения, например добавлены нестандартные режимы игры. Нестандартные режимы вводятся для того, чтобы обучающийся не мог найти готовую реализацию задания в Интернете.

К разработке программного продукта предъявляются следующие требования:

– язык программирования – C++, использование других языков **не допускается**;

– среда разработки – Visual Studio или ее аналоги;

– в разрабатываемом программном продукте **обязательно должна использоваться** одна или несколько библиотек для языка C++ по работе с графикой, окнами и звуком в Windows. В качестве такой библиотеки рекомендуется TX Library, однако допускается и использование аналогов. Последнюю версию библиотеки TX Library можно взять у преподавателя или самостоятельно скачать по ссылке http://storage.ded32.net.ru/Lib/TX/TXUpdate/Doc/HTML.ru;

– в разрабатываемом программном продукте **обязательно должен использоваться** хотя бы один паттерн проектирования;

– разрабатываемый программный продукт **должен состоять минимум из трех модулей** (использованные библиотеки также считаются модулями).

Каждый собственноручно разрабатываемый модуль должен включать минимум 3 класса;

– разрабатываемый программный продукт должен иметь модуль визуализации, работающий в графическом оконном режиме. В большинстве вариантов визуализация также предполагает анимацию (моделирования, расчетов, игрового процесса и т.п.).

4 Основные требования к оформлению пояснительной записки

Пояснительная записка является основным сопроводительным документом к выполненной работе. К оформлению пояснительной записки приступают после завершения работы над основным заданием.

Общая структура пояснительной записки состоит из титульного листа (см. приложение А), реферата с большим штампом (см. приложение В), содержания, введения, основных разделов, заключения, списка используемых источников, приложений. Основные разделы пояснительной записки содержат информацию о выполненной работе в соответствии с техническим заданием (см. приложение Б). Приложения включают подписанное обучающимся и преподавателем техническое задание, листинги основных файлов программного кода, экранные формы, графическую часть. **Изменять бланк технического задания без согласования с преподавателем запрещается. Пояснительная записка должна иметь объем не менее 30 листов формата А4 без учета приложений.**

Пояснительная записка должна быть оформлена в соответствии с требованиями, предъявляемыми к оформлению научной и технической документации, которые подробно описаны далее в этом разделе.

4.1 Общие требования

Материал, включаемый в пояснительную записку, должен быть обработан и систематизирован. Общими требованиями к оформлению пояснительной записки являются:

– четкость и логическая последовательность изложения материала;

– убедительность аргументации;

– краткость и точность формулировок, исключающих возможность неоднозначного толкования;

– конкретность изложения результатов работы;

– доказательность выводов и обоснованность рекомендаций.

В тексте пояснительной записки не допускается использование личных обращений и отдельных глаголов, подразумевающих такие обращения. Весь текст должен быть написан от третьего лица. Например, «Я разработал алгоритм», «Разработал алгоритм», «Вы видите, что...», «Видите, что...», «Мы можем утверждать, что...», «Можем утверждать, что...» и т.п. фразы некорректные. Вместо них следует использовать формулировки «Был разработан алгоритм», «Видно, что...», «Можно утверждать, что...». Также не допускается начинать предложение с союза. Например фраза «Был выполнен анализ существующего алгоритма. А также его модификация для...» некорректна. Вместо этого следует писать «Был выполнен анализ существующего алгоритма. На основе этого анализа предложена модификация для...».

Текст пояснительной записки делится на разделы (первый уровень), подразделы (второй уровень), пункты (третий уровень), подпункты (четвертый и последующие уровни). Каждый раздел следует начинать с нового листа. Пояснительная записка может содержать формулы, графики, схемы, таблицы, расчеты, приложения и другой иллюстративный материал.

Текст пояснительной записки должен быть тщательно отредактирован и набран на компьютере в текстовом редакторе. При оформлении используются следующие параметры:

– формат страницы – А4;

– ориентация – книжная (допускается размещение отдельных таблиц, рисунков на листе альбомной ориентации);

– гарнитура шрифта – Times New Roman, размер – 14 пунктов;

– поля: верхнее, нижнее – 1.5 – 2.0 см, левое – 3 см, правое – 1–1.5 см;

– межстрочный интервал – полуторный; между заголовком и текстом – множитель 3; между заголовками раздела и подраздела – множитель 2;

– абзацный отступ – 1.5 см;

– выравнивание текста в абзаце – по ширине;

– расстановка переносов – автоматическая.

Номер страницы проставляется в правом верхнем углу без точки, от края до колонтитула – 14 мм, размер колонцифры – 12 пунктов. На титульном листе, реферате, содержании номера страниц не ставятся, но учитываются в общей сквозной нумерации. Рисунки и таблицы, расположенные на отдельных листах, включаются в общую нумерацию страниц работы.

Набор в пределах всего текста должен быть единообразным по выбору шрифтов: гарнитура – Times New Roman.

4.2 Требования к структурных элементам пояснительной записки

Реферат

Реферат должен отражать количество страниц, рисунков, таблиц, приложений, использованных источников, тему, краткую характеристику работы, полученные результаты, область применения, возможность практической реализации, ключевые слова. Объем реферата должен составлять не более 1 страницы печатного текста. В самом начале после заголовка «РЕФЕРАТ» идет строка со сведениями о количестве страниц, рисунков, таблиц, приложений, используемых источников. Далее идет текстовое описание темы, краткой характеристики работы и т.п. Обычно такое описание занимает несколько небольших абзацев. В конце реферата приводится перечень ключевых слов. Ключевые слова – это основные термины, названия технологий, которые используются в работе. Всего в реферате должно быть 7-8 ключевых слов. Ключевые слова должны идти после основного текста реферата. Они оформляются прописными буквами с выравниванием по центру. Сам реферат заключается в стандартную рамку со штампом (см. рис. 4.1). Пример реферата приведен в приложении В.

В левой части штампа вносятся ФИО студента, руководителей (для курсового проектирования обычно назначают три руководителя, но на практике их количество может быть сокращено до двух – лектор и проработчик, – поэтому необходимо оставить дополнительную пустую строку для третьего руководителя), нормоконтролера. Нормоконтроллер – руководитель, который следит за качеством оформления письменных материалов, сопровождающих работу. В курсовом проектировании нормоконтроллером обычно является основной руководитель (лектор курса).

<table>
<tr><td colspan="5"></td><td colspan="4" rowspan="2">09.03.02.КР2022.18\6861.00.00 ПЗ</td></tr>
<tr><td colspan="5"></td></tr>
<tr><td>Изм.</td><td>Лист</td><td>№ документа</td><td>Подпись</td><td>Дата</td><td colspan="4" rowspan="2"></td></tr>
<tr><td>Студент</td><td colspan="2">Иванов И.И.</td><td></td><td></td><td colspan="2" rowspan="5">Web-портал интернет магазина с онлайн консультацией</td><td>Лит</td><td>Лист</td><td>Листов</td></tr>
<tr><td>Руков.</td><td colspan="2">Павлий В.А.</td><td></td><td></td><td></td><td>1</td><td>1</td></tr>
<tr><td>Руков.</td><td colspan="2">Бабакина А.А.</td><td></td><td></td><td colspan="3" rowspan="3">ДонНТУ, ФИСТ,
кафедра КМД, гр.МИД-18б</td></tr>
<tr><td></td><td colspan="2"></td><td></td><td></td></tr>
<tr><td>Н.контр.</td><td colspan="2">Павлий В.А.</td><td></td><td></td></tr>
</table>

Рисунок 4.1. Пример штампа

В правой части штампа указывается шифр и тема курсовой работы (или курсового проекта) название ВУЗа, факультета, кафедры, учебной группы.

Шифр состоит из:

– номера направления. На рис. 4.1 – «09.03.02.»;

– вида работы (курсовая работа или курсовой проект). На рис. 4.1 – «КР» (может быть «КП»);

– года разработки. На рис. 4.1 – «2022.».

– номера зачетной книжки. На рис. 4.1 – «18\6861.»

– номера ведомости. На рис. 4.1 – «00.00»

– типа документа (пояснительная записка, выпускная работа). На рис. 4.1 – «ПЗ» (может быть ВР).

Содержание

В самом начале содержания идет заголовок «СОДЕРЖАНИЕ», после которого приводятся все заголовки разделов и подразделов пояснительной записки, начиная с введения, и указываются страницы, с которых они начинаются. Заголовки содержания должны точно повторять заголовки в тексте. Последнее слово каждого заголовка соединяют отточием с соответствующим ему номером страницы в правом столбце содержания. Названия пунктов, подпунктов в содержание студенческих работ обычно не выносят.

Введение

Во введении обосновывается актуальность темы выполненной работы, определяется ее цель, формулируются задачи, которые необходимо решить, научная и/или практическая значимость, указываются сведения об общей структуре работы.

Объем введения составляет 2–4 страницы. В соответствии с этим в его структуре рекомендуется выделять соответствующие подпункты:

- актуальность;
- цель и задачи работы;
- структура работы.

При необходимости в структуру введения можно включать дополнительные подпункты.

Основная часть

Материалы пояснительной записки излагаются логически последовательно с плавным переходом от одного элемента текста к другому и связываются по

содержанию единством общего плана работы. При этом каждый раздел должен начинаться вводными словами и заканчиваться выводами и предпосылками для рассмотрения последующего материала.

Текст пояснительной записки обязательно должен содержать иллюстративный материал, который может состоять из поясняющих рисунков, схем, таблиц, блок-схем используемых или разработанных алгоритмов и т.д. Выбор методики (алгоритма) того или иного расчета, принимаемые решения должны кратко, но убедительно обосновываться.

Не рекомендуется доказывать общеизвестные и очевидные положения, а также повторять однотипные расчеты. Одни и те же фрагменты текста не могут повторяться в работе несколько раз.

Заключение

В заключении даются выводы и обобщения по выполненной работе в целом, которые включают в себя наиболее важные выводы по всем разделам. Выводы должны строго соответствовать задачам работы, сформулированным во введении, отражать практическую ценность тех результатов, к которым пришел автор. В заключении также даются рекомендации, указываются пути дальнейших исследований в рамках рассматриваемой проблемы.

Список используемых источников и ссылок на них

В список использованных источников допускается включать не только издания, которые были фактически использованы автором, но и названия работ, отвечающих тематике выполненной работы, по которым автор проводил обзор. При написании работы требуется давать ссылки на источник, из которого были заимствованы материалы, цитированы отдельные положения или использованы результаты. Для этого в квадратных скобках указывается соответствующий ему порядковый номер в списке использованных источников. Можно сослаться сразу на несколько источников, разделяя их номера в квадратных скобках запятыми. Наконец, можно указывать номер страницы источника. Для этого после номера через запятую указывается сокращение «стр.», после которого уже идет номер страницы. Помимо печатных изданий на русском языке могут использоваться материалы на иностранных языках и электронные ресурсы, при указании которых необходимо соблюдать следующий порядок:

– печатные издания на русском языке;
– печатные издания на иностранных языках;
– электронные ресурсы на русском языке;
– электронные ресурсы на иностранных языках.

Внутри каждого из перечисленных блоков списка использованных источников источники оформляются в алфавитном порядке по фамилии первого автора либо по названию (при отсутствии авторов).

В список использованных источников должно быть включено не менее 5 источников.

Каждый источник в списке использованных источников оформляется в соответствии с требованиями ГОСТ Р 7.0.100-2018 «Библиографическая запись. Библиографическое описание. Общие требования и правила составления». Примеры библиографических описаний представлены в приложении Г.

Приложения

Перед первым приложением вставляется титульный лист, на котором прописными буквами по центру листа пишется слово «ПРИЛОЖЕНИЯ». Номер на этом листе не ставится, в общую нумерацию он не включается. После этого листа следуют все приложения по порядку их упоминания в работе. Нумерация приложений производится заглавными русскими буквами. Каждое новое приложение начинается с новой страницы и имеет свое название. Первой строкой прописными буквами с выравниванием по центру записывается слово «ПРИЛОЖЕНИЕ», за которым следует его порядковый номер (соответствующая прописная буква русского алфавита). Ниже через 1.5 интервала приводится название приложения.

4.3 Требования к оформлению пояснительной записки

Оформление заголовков разделов, подразделов, пунктов

Заголовки разделов, подразделов и пунктов печатаются с абзацного отступа без точки в конце. Для разделов используются прописные буквы, для подразделов – строчные, начиная с прописной буквы. Переносы слов в заголовках не допускаются. Если заголовок состоит из двух предложений, то они разделяются точкой. Слова «Раздел», «Подраздел» в заголовке раздела и подраздела не пишутся, ставится лишь его номер без точки и приводится название.

Нумеровать разделы, подразделы и пункты следует арабскими цифрами. Разделы должны иметь порядковую нумерацию в пределах всего текста, за исключением содержания, введения, заключения, списка использованных источников, которые не нумеруются, например: 1, 2, 3 и т.д.

Номер подраздела включает в себя номер раздела и порядковый номер подраздела, между которыми ставится точка, например: 1.1, 1.2, 1.3, 1.4 и т.д. Номер пункта состоит из номера раздела, подраздела и порядкового номера пункта, отделенных точками, например: 1.1.1, 1.1.2, 1.1.3 и т.д.

После номера раздела, подраздела и пункта точка не ставится. Если раздел или подраздел имеет только один пункт, то нумеровать и выносить в содержание его не нужно.

Заголовки разделов и подразделов выравниваются по ширине (отступ 1.5 см). Заголовки «ВВЕДЕНИЕ», «ЗАКЛЮЧЕНИЕ», «СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ», «ПРИЛОЖЕНИЯ» выравниваются по центру.

Интервал после заголовка раздела равен множителю 2, между заголовком подраздела и предшествующим (последующим) текстом – множителю 3. Расстояние до и после строки заголовка пунктов принимают таким же, как в тексте.

Оформление списков

В тексте пояснительной записки могут быть приведены перечисления (списки). Перед каждым перечислением следует ставить дефис. Каждый пункт перечисления отделяется от следующего точкой с запятой, если начинается со строчной буквы, и точкой – в случае написания пунктов с прописной буквы. Последний пункт перечисления заканчивается точкой (см. рис. 4.2).

Для того чтобы избежать поражения электрическим током, пользователям запрещается:

- работать влажными и грязными руками;
- заходить в рабочую зону (за компьютеры);
- трогать разъемы соединительных кабелей;
- прикасаться к питающим проводам и устройствам заземления;
- прикасаться к экрану и к тыльной стороне монитора, клавиатуры;
- включать и отключать аппаратуру без указания преподавателя;
- самостоятельно устранять неисправности в работе аппаратуры.

Рисунок 4.2. Пример оформления списка

При оформлении разных уровней элементов списка следует соблюдать отступы: дефис перечисления начинается в той позиции, с которой начинается текст элемента предыдущего уровня. Если элемент перечисления занимает несколько строк, то отступы соблюдаются только для первой строки, остальные строки перечисления отступов не имеют.

Оформление рисунков

Рисунки (чертежи, графики, схемы, диаграммы, фотоснимки) следует располагать непосредственно после текста, в котором они упоминаются впервые, или на следующей странице. На все рисунки должны быть даны ссылки по тексту записки, например «см. рис. 3.1» или «на рис. 3.1 показана ...».

Подпись к рисунку включает его номер и название. Рисунки следует нумеровать в пределах раздела, т.е. номер рисунка должен состоять из номера раздела и порядкового номера рисунка в разделе, которые разделены точкой и с точкой в конце, например: «Рисунок 1.1.». Название рисунка отделяется от номера пробелом, после названия точка не ставится. Подпись к рисунку имеет размер 14 пунктов и располагается непосредственно под рисунком посередине строки, после которой делается отступ в одну пустую строку. Пример оформления рисунка показан на рис. 4.3.

Рисунок 3.3. Диаграмма вариантов использования

Рисунок 4.3. Пример оформления рисунка

Если название рисунка не помещается в одну строку, то его разбивают исходя из смыслового контекста на несколько строк, устанавливая одинарный межстрочный интервал. Переносы текста в этом случае недопустимы.

Рисунки в приложениях обозначают аналогичным образом, но вместо номера раздела указывается буквенное обозначение приложения, например: «Рисунок А.3.».

Если рисунок состоит из нескольких частей, то каждая из таких частей нумеруется строчной буквой русского алфавита, после которой ставится круглая скобка. При этом под названием рисунка размещается подрисуночная подпись с пояснениями по каждой части рисунка, после названия же в этом случае ставится двоеточие.

В тексте пояснительной записки не допускается расположение несколько подряд идущих рисунков, не разделенных текстом. Текст введения и заключения рисунков содержать не должен. Рамка вокруг рисунка не допускается. При оформлении блок-схем их основной фон должен быть белым.

Оформление таблиц

Таблицы применяются для большей наглядности и удобства сравнения показателей. Название таблицы должно отражать ее содержание, быть точным, кратким. Название таблицы следует помещать над таблицей справа, размер шрифта – 14 пт., без абзацного отступа в одну строку с ее номером и названием через точку, но без точки в конце (см. рис. 4.4).

Таблица 3.1. Описание классов

Класс	Атрибуты класса	Операции класса
Авторизация	Возраст	Переход к обучающему контенту
Игра	Вопрос Ответ Подсказка	Вывод задания Ответ пользователя на поставленную задачу
Теоретическая информация	Текст Параметры шрифта	Просмотр обобщающей информации
Подсказки	Номер задания Текст задания Правильный ответ	Просмотр правильного ответа к заданию
Результат	Вывод соотношения правильных и не правильных ответов	Просмотр результата тестирования

Рисунок 4.4. Пример оформления таблицы на одной странице

Если таблица не помещается на одной странице, то ее переносят. При переносе части таблицы название помещается только над первой частью таблицы, нижняя горизонтальная черта, ограничивающая таблицу, не проводится. Над последующими частями записывается фраза «Продолжение таблицы» с выравниванием текста по правому краю и указывается номер таблицы, например: «Продолжение таблицы 3.1», как показано на рисунке 4.5.

Таблица 3.2. Тип текстовых обложек с наличием противоречий

Тип текстовой обложки	Название	CTR для облжек	Наличие противоречия
с наличием четырёх и более слов	ТОП 5 СИТХОВ котор	7,4	
	Как создать армию сит	3,8	
	Нам покажут НОВОГО	2,2	
	ЙОДА и Первый орден	3,9	
	Откуда у Кайло Рена	9	+
	Если бы ГЕНЕРАЛ	7,8	+
	Почему людей больше	6,3	
	Как на самом деле Дарт	6,1	

Продолжение таблицы 3.2

Тип текстовой обложки	Название	CTR для облжек	Наличие противоречия
	Как на самом деле Дарт	6,1	

Рисунок 4.5. Пример оформления таблицы на нескольких страницах

Если таблица занимает более двух страниц, то на странице с ее окончанием вместо слова «Продолжение» записывается слово «Окончание», при этом также соблюдается выравнивание текста по правому краю.

Таблицу следует располагать непосредственно после текста, в котором она упоминается впервые, или на следующей странице. На все таблицы должны быть даны ссылки по тексту записки, например «см. табл. 3.1» или «в табл. 3.1 показана ...». Таблицу с большим количеством колонок допускается размещать на листах в альбомной ориентации, однако номера страниц этих листов должны быть как и во всей записке. т.е. в правом верхнем углу листа портретной ориентации. Часто для этого приходится печатать таблицу отдельно, а номер страницы отдельно, так как в Word повернуть номер отдельной страницы довольно непросто.

Если данные в какой-либо ячейке таблицы не приводятся, то в ней ставится прочерк.

Таблицы, за исключением таблиц приложений, следует нумеровать в пределах раздела, как и рисунки. В этом случае номер таблицы состоит из номера раздела и порядкового номера таблицы, которые разделены точкой.

Номер таблицы каждого приложения обозначается арабскими цифрами с добавлением перед цифрой обозначения приложения. Заголовки колонок и строк таблицы следует писать с прописной буквы в единственном числе, подзаголовки колонок – со строчной буквы, если они составляют одно предложение

с заголовком, или с прописной буквы, если они имеют самостоятельное значение. В конце заголовков и подзаголовков таблиц точки не ставятся.

Таблицы слева, справа и снизу, как правило, ограничены линиями. В таблице применяют размер шрифта меньше на 2 пункта, чем в основном тексте, например 12 пунктов при размере основного текста 14 пунктов. Набор текста, содержащегося в таблице, выполняется через одинарный интервал. Разделять заголовки и подзаголовки боковика и колонок диагональными линиями не допускается.

Горизонтальные и вертикальные линии, разграничивающие строки таблицы, допускается не проводить, если их отсутствие не затрудняет пользование таблицей. Заголовки колонок, как правило, записываются параллельно строкам таблицы. При необходимости допускается перпендикулярное расположение заголовков колонок с центрированием относительно горизонтали и вертикали.

Тексты введения и заключения не должны содержать таблиц.

Оформление математических формул

Математические формулы представляют собой символическую запись некоторого высказывания или суждения. Они отличаются по характеру составляющих их элементов, а также по числу занимаемых ими строк (однострочные, двустрочные, многострочные).

Математические формулы и уравнения, вынесенные в отдельные строки, как правило, располагаются посередине строки. Если формула не вмещается в одну строку, то она должна быть перенесена после знаков равенства «=», плюса «+», минуса «–», умножения «·» или деления «:», причем знак в начале следующей строки повторяется. При переносе формулы на знаке, символизирующем операцию умножения, применяется символ «×». Поскольку в формулах часто фигурируют нестандартные символьные обозначения, например, греческие буквы, специальные математические знаки, то формулы удобнее всего набирать, используя специализированный редактор Microsoft Equation или Math Type. Первый обычно уже встроен в Word (до 2010 версии включительно), а второй необходимо устанавливать отдельно. Создать новую формулу можно при помощи меню «Вставка» – «Объект» – «Microsoft Equation 3.0», отредактировать существующую – при помощи двойного щелчка на формуле.

Пояснения символов и коэффициентов, входящих в формулу, если они не сделаны ранее в тексте, должны быть помещены непосредственно под формулой. Пояснения каждого символа следует давать с новой строки в той последовательности, в которой символы приведены в формуле. Первая строка пояснения должна начинаться со слова «где» без двоеточия после него и без абзацного отступа в начале строки (см. рис. 4.6).

В конце формулы и в тексте перед ней знаки препинания ставятся в соответствии с правилами пунктуации. Двоеточие ставится, если в тексте перед формулой содержится обобщающее слово. При указании ссылки на формулы в основном тексте работы формулы следует в пределах раздела, используя двухуровневую нумерацию арабскими цифрами, заключенными в круглые скобки, которая состоит из номера раздела и порядкового номера формулы, разделенных точкой, например: «(7.1)», и выравнивать по правому краю строки.

Для вычисления воздухообмена по теплу используется формула:

$$L = \frac{Q_{изб.}}{c*p*(t_в - t_н)} \quad (7.1)$$

где $Q_{изб}$ – избыточное выделение тепла в рабочем помещении, ккал / ч;

с – теплоемкость воздуха (0,239 ккал/кг°С);

р – плотность воздуха (1,205 кг/ $м^3$);

$t_в$ – температура внутри помещения (30°С);

$t_{пр}$ – температура снаружи (22°с).

Необходимо рассчитать избыточное поступление тепла по формуле:

$$Q_{изб} = Q_{обор} + Q_л + Q_{осв} + Q_{рад} \quad (7.2)$$

где $Q_{обор}$ – выделение тепла от электронногооборудования;

$Q_л$ – выделение тепла учениками, занимающихся в классе;

Рисунок 4.6. Оформление формул в тексте

Группа формул, объединенных фигурной скобкой, имеет один номер. Ссылки в тексте на порядковые номера формул даются в скобках, например: «...подставив в формулу (7.1) выражение ...». Формулы, помещенные в приложениях, должны нумероваться отдельно арабскими цифрами в пределах каждого приложения с добавлением перед каждой цифрой обозначения приложения, например: «(В.1)».

Оформление программного кода

При необходимости представления в тексте пояснительной записки программного кода соответствующий блок кода предваряется надписью. Надпись включает слово «Листинг», его номер и название. Блоки кода следует нумеровать в пределах раздела, т.е. номер должен состоять из номера раздела и порядкового номера блока кода в разделе, которые разделены точкой и с точкой в конце, например: «Листинг 1.1.». После номера блока кода приводится его название, начинающееся с прописной буквы, без точки в конце. Выравнивается надпись как и сам код по левому краю.

Программный код должен снабжаться достаточным количеством поясняющих комментариев. **Для текста программного кода используется шрифт Courier New, размер шрифта 10-12 пунктов. Межстрочный интервал – одинарный.**

Допускается использовать цветную разметку программного кода с выделением ключевых слов, однако цвета должны быть подобраны так, чтобы при печати в черно-белом режиме все участки блока кода были читабельны. После приведенного кода делается отступ в одну строку.

Если строка программного кода не помещается в одну строку, то она **должна быть отформатирована** так, чтобы это не противоречило правилам и синтаксису приводимого программного кода. На практике это означает, что

строки должны разбиваться и переноситься без нарушения синтаксиса программного кода. На рис. 4.7. показан типичный пример неправильного форматирования, которое получается после копирования исходного кода из среды программирования в Word.

```
        // Метод запускает анимацию
        public void startAnimation(int freq, int angle, int ax, int ay, int
bx, int by, int cx, int cy) {
            // Сохранить параметры
            this.angl = 0;
            this.ax = ax;
            this.ay = ay;
            this.bx = bx;
            this.by = by;
            this.cx = cx;
            this.cy = cy;
            // Вычислить координаты центра тяжести треугольника
            this.mx = (this.ax + this.bx + this.cx) / 3.0;
            this.my = (this.ay + this.by + this.cy) / 3.0;
            // Создать таймер
            this.tmr = new Timer();
            // Здесь создается экземпляр анонимного класса, наследованного от
TimerTask и в нем
            // переопределяется метод run. Этот метод будет вызываться с
указанной периодичностью 1000/freq
            this.tmr.schedule(new TimerTask() {
                @Override
                public void run() {
                    angl = (angl + angle) % 360;
                    draw();
                }
            }, 0, 1000/freq);
        }
```

Рисунок 4.7. Неправильное оформление программного кода

На рисунке 4.8 показан тот же код, но уже отформатированный для публикации на листе формата А4. Легко видеть, что длинные комментарии теперь разбиты на несколько строк, а список аргументов функции был перенесен на другую строку, не нарушая при этом синтаксиса приводимого кода.

```
    // Метод запускает анимацию
    public void startAnimation(int freq, int angle, int ax, int ay,
                               int bx, int by, int cx, int cy) {
        // Сохранить параметры
        this.angl = 0;
        this.ax = ax;
        this.ay = ay;
        this.bx = bx;
        this.by = by;
        this.cx = cx;
        this.cy = cy;
        // Вычислить координаты центра тяжести треугольника
        this.mx = (this.ax + this.bx + this.cx) / 3.0;
        this.my = (this.ay + this.by + this.cy) / 3.0;
        // Создать таймер
        this.tmr = new Timer();
        // Здесь создается экземпляр анонимного класса, наследованного
        // от TimerTask и в нем
        // переопределяется метод run. Этот метод будет вызываться с
        // указанной периодичностью
        // 1000/freq
        this.tmr.schedule(new TimerTask() {
            @Override
            public void run() {
                angl = (angl + angle) % 360;
                draw();
            }
        }, 0, 1000/freq);
    }
```

Рисунок 4.8. Правильное оформление программного кода

В результате читабельность кода резко возрастает, но и сам код при этом остается работоспособным и может быть снова скопирован в среду программирования для внесения доработок или компиляции.

Не допускается обводить исходный код рамкой или печатать его на темном фоне. Последнее приводит к огромному расходу краски при печати.

Не допускается приводить программный код целиком в основных разделах пояснительной записки, ввиду его большого объема, но можно приводить отдельные фрагменты (части) кода, перемежая их текстом, описывающим работу отдельных частей алгоритма. Полная версия программного кода приводится только в приложении.

Не допускается оформлять программный код или его фрагменты в несколько столбцов. Это также не способствует читабельности.

Код должен быть логически отформатирован, т.е. вложенные блоки (например тело цикла) должны иметь отступ слева в 3-4 пробела.

5 Примерное содержание пояснительной записки

Пояснительная записка состоит из титульного листа, реферата, содержания, введения, основных разделов, заключения, списка используемых источников, приложений. Основные правила оформления указанных пунктов были рассмотрены в предыдущем разделе. Далее указанные пункты будут рассмотрены с точки зрения содержания.

Образец **титульного листа** и реферата приведены в приложениях А и В соответственно. **В реферате** необходимо привести сведения о количестве страниц, рисунков, таблиц, приложений, используемых источников. Далее следует кратко – по одному предложению – охарактеризовать тему и цель работы, объект разработки, перечислить программные средства, которые использовались для разработки, полученные результаты. Наконец, следует привести перечень ключевых слов или используемых понятий. Объем реферата – не более одной страницы.

Во введении требуется охарактеризовать актуальность разработки, в отличие от реферата более детально представить тему и цель работы, объект разработки, дать описание структуры пояснительной записки (из каких разделов состоит и что в этих разделах рассматривается). Объем введения – не более двух страниц.

Далее следуют основные разделы пояснительной записки. При разработке этих разделов следует придерживаться рекомендуемых названий. Предполагаемое содержимое каждого раздела также должно соблюдаться.

1. **Постановка задачи.**

В данном разделе необходимо привести полный текст задания на разработку (можно скопировать из варианта задания), а также его предварительный анализ. Например, «Предполагается, что в ходе выполнения курсового проекта будет разработано программное приложение, которое...», «Для реализации [такой-то возможности] потребуется разработать [такую-то структуру данных]». На данном этапе не следует описывать конкретные классы или модули, этот раздел должен давать лишь общее представление о работе, которая будет проделана.

2. **Выбор инструментария для разработки.**

В данном разделе необходимо привести 2-3 примера существующих инструментов разработки программного кода на C++ (допускается описывать как интегрированные среды разработки, так и усовершенствованные редакторы программного кода, поддерживающего C++). После обзора инструментов необходимо составить сравнительную таблицу их возможностей, на основании которой выбрать подходящий для дальнейшей разработки инструмент, т.е. провести анализ. Например, после таблицы с обзором возможностей можно написать следующую фразу «В результате анализа таблицы [такой-то] можно видеть, что [такой-то инструмент] обладает [такими-то возможностями], которые необходимы при разработке. Поэтому в дальнейшем для разработки будет использован [именно этот инструмент или именно эта интегрированная среда разработки]».

3. Проектирование программного продукта.

В данном разделе необходимо попытаться выделить основные модули будущего программного продукта и определить связи между ними, представить схему взаимодействия модулей (в т.ч. сторонних библиотек), описав назначение каждого модуля на схеме. Также в данном разделе следует привести описание основных структуры данных, если они есть. Наконец, следует обязательно словесно пояснить, какой паттерн проектирования предполагается использовать и для чего.

4. Объектно-ориентированный анализ программного продукта.

В данном разделе следует спроектировать и описать классы, реализующие функционал каждого из модулей. Описывать функционал классов следует раздельно по каждому модулю. Сначала представляется схема связей классов (т.н. диаграмма классов), затем следует описать назначение и функционал каждого класса, перечислить свойства, методы, конструкторы, деструкторы классов, дать описание каждого свойства.

Поскольку обучающиеся ранее не сталкивались с понятием «диаграмма классов» (более подробно эта тема изучается на старших курсах), следует дать здесь краткое пояснение.

Определение: Диаграмма классов – структурная диаграмма языка моделирования UML, демонстрирующая общую структуру иерархии классов системы, их кооперaций, атрибутов (полей), методов, интерфейсов и взаимосвязей (отношений) между ними [Википедия].

На диаграмме классов каждый класс представляется прямоугольником, в верхней части которого по центру указывается название класса. Затем, после горизонтальной линии следует перечень свойств. Для каждого свойства указывается модификатор доступа, название свойства, тип и начальное значение (если есть). После очередной горизонтальной черты аналогично указывается перечень методов. Модификаторы доступа обозначаются на диаграмме следующим образом:

– + Публичный (Public);
– - Приватный (Private);
– # Защищённый (Protected).

Взаимосвязи между объектами классов показываются на диаграмме стрелками. Взаимосвязи могут иметь различные типы, однако наиболее распространенным считается наследование класса и агрегация. На рис. 5.1 показаны все возможные варианты взаимодействий, но следует знать, что некоторые типы взаимодействия (например реализация) невозможно реализовать на C++.

Рис. 5.1. Обозначения взаимосвязей на диаграммах классов

Ассоциация показывает, что объекты одной сущности (класса) связаны с объектами другой сущности таким образом, что можно перемещаться от объектов одного класса к другому. Является общим случаем композиции и агрегации.

Агрегация – это разновидность ассоциации при отношении между целым и его частями. Агрегация встречается, когда один класс является коллекцией или контейнером для других.

Композиция – более строгий вариант агрегации. Она имеет жёсткую зависимость времени существования экземпляров класса контейнера и экземпляров содержащихся классов. Если контейнер будет уничтожен, то всё его содержимое будет также уничтожено.

Наследование показывает, что один из двух связанных классов является частной формой другого.

Реализация – отношение между двумя элементами модели, в котором один элемент реализует поведение, заданное другим элементом, например абстрактным классом или интерфейсом.

Зависимость – это слабая форма отношения использования, при которой изменение в спецификации одного влечёт за собой изменение другого, причём обратное не обязательно. Возникает, когда объект выступает, например, в форме параметра или локальной переменной.

Пример диаграммы классов показан на рис. 5.2. Глядя на эту или подобную диаграмму, программист в состоянии представить себе структуру реализуемого модуля или проекта в целом.

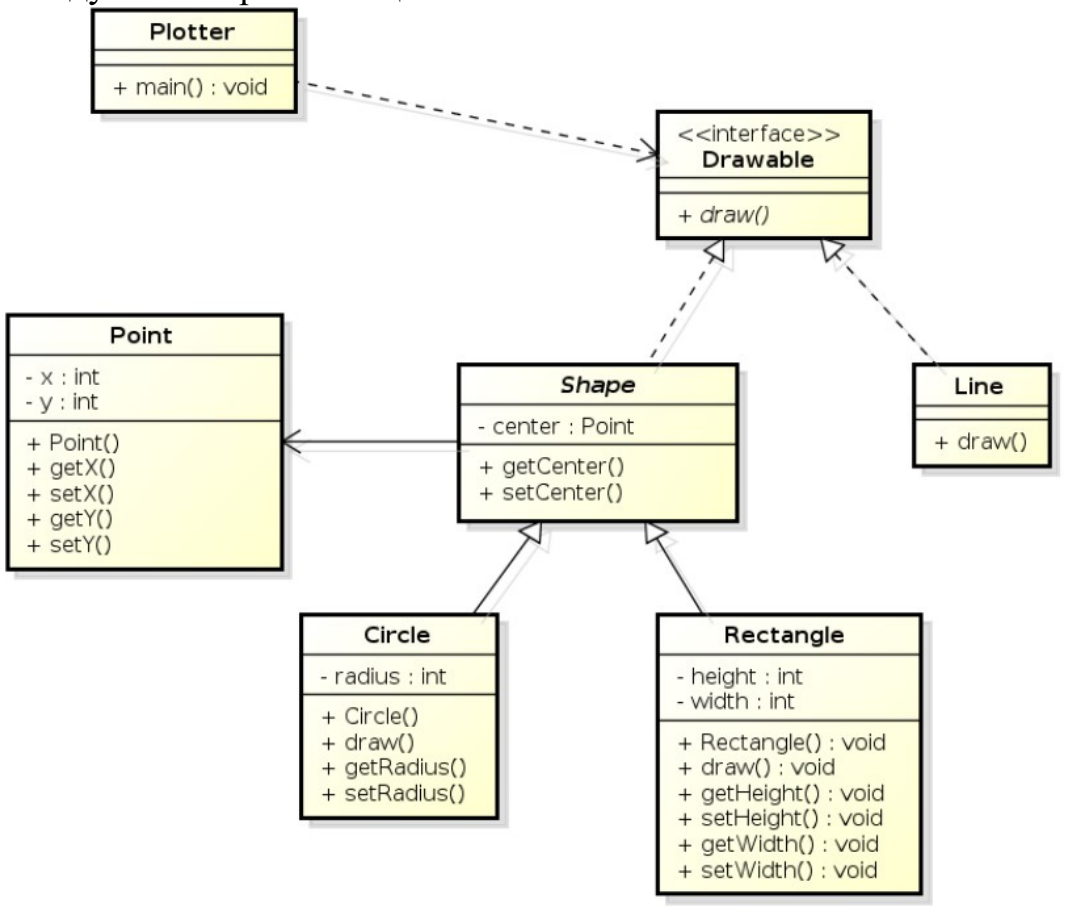


Рис. 5.2. Пример диаграммы классов

Следует заметить, что некоторые интегрированные среды разработки позволяют строить подобные диаграммы автоматически по разработанному коду. В Visual Studio, например, такая возможность есть.

Диаграмма классов в тексте раздела не приводится, а выносится в графическую часть проекта. По тексту необходимо предусмотреть ссылки на эту диаграмму, например, «Диаграмма классов представлена на рис. В.1.».

5. Разработка программного продукта.

В данном разделе следует описать алгоритмы работы основных методов основных классов при помощи блок-схем. При создании блок-схем допускается не использовать детальное представление (вплоть до операторов языка), а использовать укрупненные блоки, словесно описывая, как они работают. Описывать работу классов, представляющих какой-либо паттерн, не нужно. Блок-схемы в тексте не приводятся, а выносятся в графическую часть проекта. По тексту необходимо предусмотреть ссылки на эту диаграмму, например, «Блок-схема работы модуля [такого-то] представлена на рис. В.2.».

6. Описание программного продукта.

В данном разделе следует привести описание программного продукта с точки зрения пользователя – основные функции, внешний вид интерфейса. Скриншоты работы программного продукта также выносятся в приложение, которое следует за графической частью.

Объем основных разделов суммарно должен составлять не менее 25 страниц формата А4.

В заключении следует сделать выводы и обобщения по выполненной работе в целом, которые включают в себя наиболее важные выводы по всем разделам. Объем заключения – не более одной страницы.

В списке использованных источников приводится не менее 5 источников, оформленных в соответствии с ГОСТ Р 7.0.100-2018 «Библиографическая запись. Библиографическое описание. Общие требования и правила составления». Объем списка – не более одной страницы, примеры библиографических описаний приведены в приложении Г.

Приложения к курсовому проекту включают:

А) Техническое задание (бланк из приложения Б необходимо распечатать, заполнить и подписать, затем принести на подпись преподавателю).

Б) Листинг программы.

В) Графическая часть проекта. Включает схему взаимодействия модулей, диаграмму классов, блок-схемы, отражающие работу основных алгоритмов, разработанные структуры данных (при необходимости). Каждая схема или диаграмма содержит большой штамп и рамку (как в реферате, см. рис. 4.1).

Г) Экранные формы. Здесь следует привести основные скриншоты работы программного продукта. Штампы и рамки здесь не используются.

Если курсовой проект сдается дистанционно, то бланк ТЗ необходимо распечатать, заполнить, подписать, отсканировать и вставить в пояснительную записку. Сама записка присылается в формате PDF. Подпись преподавателя на ТЗ в этом случае необязательна.

6 Защита и критерии оценивания курсового проекта

Защита курсовой работы (или курсового проекта) проводится с целью оценки степени теоретической и практической подготовленности обучающегося, а также оценивания уровня подготовки к профессиональной деятельности в соответствующей области. Результатом защиты является подведение итогов выполнения работы и выставление итоговой оценки.

Защита курсовой работы (или курсового проекта) происходит перед комиссией в количестве 2-3 преподавателей, могут также присутствовать другие обучающиеся группы. В состав комиссии входят лектор курса, проработчик, а также приглашенный преподаватель, ведущий сопутствующие дисциплины. Защита начинается с доклада в ходе которого обучающийся докладывает основные положения проделанной работы. Доклад должен быть лаконичным, занимать не более 5-ти минут и отражать: актуальность, цель и задачи работы; основное содержание каждого раздела пояснительной записки и выводы по ним и работе в целом. Далее обучающийся отвечает на вопросы преподавателей. Количество задаваемых вопросов и время ответов на них не ограничено. Комиссия, в процессе обсуждения, отмечает достоинства и недостатки проделанной работы и проведенной защиты, после чего объявляется итоговая оценка.

Итоговая оценка, выставляемая по 100-бальной шкале, зависит не только от успешного выступления обучающегося, а складывается из нескольких критериев в соответствии с таблицей 6.1.

Таблица 6.1. Основные критерии оценивания выполненной работы

Критерий	**Максимальное количество баллов по 100-бальной шкале**
Разработка и демонстрация работоспособности программного кода	25
Разработка и оформление пояснительной записки	15
Доклад основных положений выполненной работы	10
Ответы на задаваемые вопросы	30
Своевременность выполнения основных этапов работы (соответствие календарному плану выполнения работы из ТЗ)	20
Итого:	**100**

Комиссия имеет право назначать количество баллов по каждому из критериев в соответствии с адекватным оцениванием выполненной работы в соответствии с таблицей 6.2.

Таблица 6.2. Основные критерии оценивания выполненной работы

Описание результата оценивания	**Количество баллов**
Разработка и демонстрация работоспособности программного кода	
Код не работоспособен	0
Код работоспособен, однако обучающийся выполнил не то задание или не свой вариант	5
Код работоспособен, однако в нем имеются существенные ошибки или замечания, например неполная реализация в соответствии с ТЗ	15

Продолжение таблицы 6.2

Код работоспособен, однако в нем имеются несущественные ошибки или замечания, например отсутствие проверок исходных данных на корректность	20
Код полностью работоспособен, выполненная работа полностью соответствует ТЗ	25
Разработка и оформление пояснительной записки	
Пояснительная записка отсутствует*	-
В пояснительной записке имеются существенные замечания, например описан проект, не соответствующий варианту задания	0
В пояснительной записке имеются существенные замечания, например представлены не все разделы, предусмотренные ТЗ	5
В пояснительной записке имеются существенные грамматические или стилистические замечания	10
В пояснительной записке имеются несущественные грамматические или стилистические замечания	13
Пояснительная записка выполнена без замечаний и в соответствии с требованиями оформления	15
Доклад основных положений выполненной работы	
Доклад не состоялся	0
Доклад выполнен с существенным превышением отведенного времени с большим количеством информации не по сути работы	5
Доклад выполнен в соответствии с требованиями, отражены все основные положения и сделаны выводы по проделанной работы	10
Ответы на задаваемые вопросы	
Обучающийся не смог ответить ни на один заданный вопрос	0
Обучающийся ответил полно примерно на 25% заданных вопросов, на остальные не ответил	10
Обучающийся ответил полно примерно на 50% заданных вопросов, на остальные не ответил	15
Обучающийся ответил полно примерно на 75% заданных вопросов, на остальные не ответил	20
Обучающийся ответил на все вопросы, однако ответы были неполными или не развернутыми	25
Обучающийся ответил на все вопросы	30
Своевременность выполнения основных этапов работы	
Основные этапы работы выполнялись несвоевременно, обучающийся не демонстрировал промежуточные этапы выполнения работы	0
Основные этапы работы выполнялись частично несвоевременно, обучающийся продемонстрировал выполнение примерно 50% основных этапов в соответствии с календарным графиком	10
Основные этапы работы выполнялись своевременно, обучающийся продемонстрировал выполнение всех основных этапов в соответствии с календарным графиком	20

* – обучающийся не допускается к защите в соответствии с «Положением об организации учебного процесса ДонНТУ» (утверждено приказом №337-14 от 02.05.18)

Если на защите обучающийся набирает менее 60 баллов из 100 максимально возможных ставится оценка «неудовлетворительно». В этом случае обучающийся имеет право на пересдачу, **причем по решению комиссии зада-**

ние может быть выдано новое. Если обучающийся получает оценку неудовлетворительно более трех раз, то решением декана факультета собирается новая комиссия в лице декана, заведующего кафедрой, лектора курса, ведущих профессоров или доцентов кафедры, в задачу которой входит установить целесообразность продолжения обучения студента на указанном направлении или профиле и его отчислении в случае принятия соответствующего решения.

В случае получения положительной оценки (от 60 до 100 баллов) эта оценка переводится в 5-бальную шкалу и шкалу ECTS в соответствии с таблицей 6.3.

Таблица 6.3. Таблица перевода оценок из 100-бальной шкалы в 5-бальную шкалу и шкалу ECTS

Сумма баллов по 100-бальной шкале	**Оценка по шкале ECTS**	**Оценка по государственной шкале**
90-100	A	Отлично
80-89	B	Хорошо
75-79	C	
70-74	D	Удовлетворительно
60-69	E	
35-59	FX	Неудовлетворительно
0-34	F*	

* – с обязательным повторным изучением дисциплины

Оценка «отлично» означает, что выполненная работа в полной мере отвечает исходным требованиям технического задания, программный код разработан корректно, сопроводительная документация выполнена грамотно и логично, а сама работа выполнена в отведенные сроки.

Оценка «хорошо» означает, что выполненная работа частично не соответствует исходным требованиям технического задания, или имеются нарушения требований в оформлении пояснительной записки.

Оценка «удовлетворительно» означает, что работа формально отвечает требованиям, однако имеются существенные недоработки исходного кода, оформления сопроводительной документации, а сама работа сдавалась не вовремя.

Оценка «неудовлетворительно» означает, что работа не выполняет поставленной задачи, пояснительная записка либо отсутствует вообще, либо оформлена с серьезными нарушениями, а сам обучающийся не смог ответить на поставленные вопросы, а работа была предоставлена к рассмотрению с существенным нарушением календарного графика.

7 Варианты заданий

7.1 Специализированные приложения

1. Нотный редактор

Нотный редактор представляет собой приложение для работы с нотами, аккордами, паузами, тактовыми чертами. Нота характеризуется тональностью, октавой, альтерацией (диез, бемоль, бекар, дубль-диез, дубль-бемоль) и продолжительностью звучания. Аккорд состоит из нескольких нот, проигрываемых одновременно, причем каждый звук в аккорде может иметь свою продолжительность. Пауза характеризуется только продолжительностью. Тактовая черта – это разделитель музыкального произведения на такты, число нот в такте определяется продолжительностью. Нотный редактор должен обеспечивать следующие функции:

– создание, удаление, перемещение, копирование нот, аккордов, пауз, тактовых черт;

– загрузка и сохранение музыкальной композиции в текстовый файл (в собственном формате или формате MID по согласованию с преподавателем);

– отображение текущей музыкальной композиции в виде нотного стана с возможностью прокручивания, если музыкальная композиция слишком большая;

– выделение аккорда по ноте, которая ему принадлежит;

– автоматическая расстановка и удаление тактовых черт;

– возможность ввода размера (числитель и знаменатель) нотного такта и отображения его на нотном стане;

– возможность ввода темпа звучания и отображения его на нотном стане;

– возможность ввода основных аккордов для ускорения ввода.

Визуально редактор состоит из панели инструментов, панели свойств для ноты, аккорда, паузы, рабочей области. В панели инструментов можно выбрать создание новой ноты, аккорда или паузы. Удаление нот, пауз и аккордов происходит путем выделения их в рабочей области на нотном стане; при этом, если нота, аккорд или пауза выделены, то на панели инструментов активируется соответствующая кнопка. Если выбирается аккорд, то все ноты в нем должны подсвечиваться, а в панели свойств отображаются их свойства без возможности редактирования. Повторный щелчок по ноте выбранного аккорда открывает свойства самой ноты с возможностью редактирования ее характеристик. Если нота одиночная, то редактирование ее характеристик через панель свойств возможно сразу после ее выбора. Если выбирается пауза, то в панели свойств показывается только продолжительность. Если выбирается тактовая черта, то панель свойств неактивна. Аналогично реализуются функции перемещения и копирования.

2. Графический редактор

Графический редактор представляет собой приложение для рисования графических примитивов – прямая, точка, отрезок, круг, эллипс, дуга, замкну-

тый полигон, ломаная, кольцо, треугольник, правильный шестиугольник. Каждую из фигур можно:

– создать;
– удалить;
– переместить;
– изменить цвет контура и заливки (только для замкнутых фигур);
– изменить толщину линии контура и ее стиль;
– изменить стиль заливки;
– скопировать;
– повернуть на произвольный угол;
– поместить на передний план.

Визуально редактор состоит из панели инструментов, панели свойств графического примитива, менеджера примитивов, рабочей области. В панели инструментов можно выбрать новую фигуру для рисования, либо отменить уже выбранную. В панели свойств можно указать параметры новой или изменить параметры существующей фигуры. В менеджере примитивов показываются все уже созданные примитивы. Выбор примитива в менеджере приводит к его выделению в рабочей области и загрузке в панель свойств параметров примитива. Кроме того, в приложении должна быть реализована возможность загрузки и сохранения списка примитивов, а также сохранение в виде изображения.

Описание функций:

– создание примитивов возможно двумя способами – путем ввода параметров в панель свойств и щелчками мыши в рабочей области. Кроме того, каждый из примитивов можно создать одним из нескольких способов. Создание прямой и отрезка возможно двумя способами – по двум точкам и по точке и углу. Т.е. для создания прямой сначала ее необходимо выбрать из панели инструментов, а далее либо заполнить поля в панели свойств и нажать «ОК», либо дважды последовательно кликнуть в рабочей области, отмечая точки на прямой или концы отрезка. Создание точки возможно одним способом. Создание круга – тремя способами (по центру и радиусу, по трем точкам, лежащим на окружности, по координатам ограничивающего квадрата). Создание эллипса – двумя способами (по центру и двум радиусам, по координатам ограничивающего квадрата). Создание дуги выполняется также, как и создание круга, с дополнительным вводом углов начала и конца. Замкнутый полигон и ломаная создаются одним способом – ввод числа вершин и задание их координат. Кольцо создается аналогично кругу с дополнительным указанием малого радиуса. Треугольник задается тремя способами (по трем точкам, по двум точкам и высоте, по двум точкам и углу). Правильный шестиугольник задается двумя способами (по координатам двух соседних вершин и по координатам центра и радиуса);

– удаление, копирование и изменение параметров примитива возможно после выбора его в менеджере;

– перемещение примитива возможно после выбора его в менеджере и ввода смещения;

– поворот на произвольный угол выполняется относительно центра примитива, для реализации использовать формулы поворота координат;

– если фигуры частично перекрывают друг друга, одну из фигур можно поместить на передний план.

3. Калькулятор матриц и векторов

Калькулятор представляет собой приложение для осуществления базовых операций над матрицами и векторами. Визуально, интерфейс представляет собой панель исходных данных, предназначенной для ввода двух матриц (A, B) и двух векторов (a, b) и числа k, панель результата, предназначенной для вывода результата операции, панель операций, на которой располагаются кнопки управления. При вводе матриц и векторов необходимо предусмотреть ввод их размерности, а также возможность автоматического заполнения случайными числами, включая отрицательные значения. Вывод результата операции должен производиться графически, в стандартных математических обозначениях и должен включать аргументы, обозначение операции и результат. Основные поддерживаемые операции:

– возврат размерности матриц A или B, векторов a или b;

– возврат элемента матриц A или B, векторов a или b по заданному индексу;

– сложение и вычитание матриц A и B, векторов a и b;

– умножение матриц A или B, векторов a или b на число k;

– умножение матриц A и B, возведение в степень k матрицы A, возведение в степень k матрицы B;

– транспонирование матриц A или B, векторов a или b;

– умножение матрицы A или B на вектор a или b;

– вычисление скалярного произведения векторов a и b;

– вычисление косинуса угла между векторами a и b;

– вычисление определителя или перманента матрицы A или B при условии, что размер матрицы не более 4;

– вычисление обратной матрицы A или B по методу Жордана;

– вычисление нормы вектора a или b и матрицы A или B;

– решение системы линейных уравнений, коэффициентами которого являются элементы матрицы A или B, а коэффициентами свободного члена – элементы вектора A или B;

– определение того, является ли матрица A или B: нулевой, единичной, симметричной, квадратной, верхнетреугольной, нижнетреугольной, кососимметричной, диагональной;

– вычисление ранга матрицы A или B.

4. Калькулятор многочленов

Калькулятор представляет собой приложение для осуществления базовых операций над многочленами. Визуально, интерфейс представляет собой панель исходных данных, предназначенной для ввода двух многочленов A и B с одной неизвестной (например: $x^2 + x + 2$) и чисел k и m, панель результата, предназначенной для вывода результата операции, панель операций, на которой располагаются кнопки управления. При вводе многочленов необходимо преду-

смотреть ввод их размерности, а также возможность автоматического заполнения случайными числами, включая отрицательные значения. Вывод многочленов должен производиться графически, в стандартных математических обозначениях и должен включать аргументы, обозначение операции и результат. Основные поддерживаемые операции:

– возврат размерности многочлена A или B;
– сложение или вычитание многочленов A или B и числа k;
– сложение или вычитание многочленов A и B;
– умножение многочленов A и B;
– возведение многочлена A или B в степень k.
– деление многочленов A и B с выдачей частного и остатка:
– поиск корней многочлена A или B при условии, что его степень не более 4. При вычислении корней многочлен считать равным нулю;
– вычисление значений многочлена A или B в заданной точке;
– дифференцирование многочлена A или B;
– вычисление неопределённого интеграла многочлена A или B;
– вычисление определённого интеграла многочлена A или B от k до m;
– проверка, является ли многочлен A или B двучленом, трехчленом;
– проверка, все ли коэффициенты многочлена A или B являются целыми числами;
– проверка, есть ли у многочлена A или B свободный член;
– построение графика многочлена A или B.

5. Калькулятор функций алгебры-логики (ФАЛ)

Калькулятор представляет собой приложение для осуществления базовых операций над ФАЛ. Визуально, интерфейс представляет собой панель исходных данных, предназначенной для ввода двух ФАЛ с возможностью указания числа переменных, панель результата, в которой будет выводиться результат операции, панель операций, на которой располагаются кнопки управления. Ввод и вывод ФАЛ должны производиться:

– в виде таблицы истинности – при вводе пользователь указывает число переменных ФАЛ, после чего появляется заготовка для заполнения таблицы истинности; при выводе – таблица истинности просто выводится на экран;
– в виде КНФ и ДНФ – ввод осуществляется путем допустимых комбинаций клавиатуры, например «!(a0|a1a2)», вывод осуществляется графически, например $\overline{a_0 \cup a_1 a_2}$.

Также необходимо предусмотреть возможность синхронного отображения вводимых ФАЛ в графическом виде для удобства восприятия.

Основные поддерживаемые операции:

– возврат размерности первой и второй ФАЛ;
– конъюнкция двух ФАЛ (только если они заданы в виде ДНФ);
– дизъюнкция двух ФАЛ (только если они заданы в виде КНФ);
– перевод из ДНФ в КНФ одной из трех ФАЛ (исходных или результирующей);

– минимизация одной из трех ФАЛ (процесс минимизации показать наглядно графически).

6. Редактор неориентированного графа

Редактор представляет собой приложение для осуществления базовых операций над графом. Визуальный интерфейс приложения состоит из области, в которой отображается граф, блока ввода графа и панели управления. Область отображения для графа позволяет увидеть сам граф и операции, выполняемые над ним. Вершины графа отображаются в виде закрашенных окружностей, связи между вершинами обозначаются линиями. Все вершины должны быть подписаны (латинские буквы от A до Z), координаты вершин задаются константами, либо выбираются случайно. У ребер, соединяющих вершины, есть веса, они тоже должны отображаться. Блок ввода графа состоит из поля для ввода количества вершин, а также зависимую от нее матрицу смежности. Матрица смежности состоит из нескольких полей, количество которых определяется количеством вершин. В этих полях указываются веса ребер, либо ноль, если ребра нет.

Панель управления представляет собой множество кнопок, по которым осуществляются базовые операции над графом. Результат выполнения операции должен отображаться на графе. Пока выполнение операции не завершится, остальные операции не должны быть доступны. Необходимо реализовать следующие операции:

– вычисление количества вершин, ребер, суммарного веса ребер;

– поиск эйлерова и гамильтонова циклов;

– поиск кратчайшего пути между заданными вершинами по алгоритму Дейкстры;

– построение минимального остовного дерева по алгоритму Прима;

– расчет степени вершин графа (для всех вершин);

– расчет максимальной пропускной способности между заданными вершинами по алгоритму Форда-Фулкерсона.

Также необходимо реализовать загрузку и сохранение графа в файл.

7. Калькулятор множеств

Калькулятор представляет собой приложение для осуществления базовых операций над множествами. Множество – это набор однотипных элементов, но разных по значению, в этом его главное отличие от одномерного массива. Множества могут задаваться в табличном виде, где каждая строка – элемент множества или в виде одной строки, разделяя элементы множества запятыми. При этом сама запятая не может быть частью элемента множества во избежание путаницы.

Визуально, интерфейс приложения представляет собой панель исходных данных, панель результата, панель операций. Панель исходных данных предназначена для ввода и отображения двух множеств A и B. Ввод множеств можно осуществлять в виде одной строки (разделитель – запятая). Отображение множества должно осуществляться в виде таблицы в графическом режиме (таблица должна быть полностью нарисована). Панель результата предназначена для вы-

вода результата одной из операций, которые могут быть проделаны над множествами. Панель операций состоит из нескольких кнопок управления. Нажатие на каждую из кнопок приводит к выполнению некоторой операции. Необходимо реализовать следующие операции:

– загрузка и сохранение множеств A и B в файл;

– проверка множеств A и B на корректность (не должно быть одинаковых значений);

– вычисление количества элементов в множествах A и B;

– сортировка элементов множества A и B по значению;

– проверка множеств на равенство $A = B$ (все элементы из множества A равны элементам из множества B);

– проверка множеств A и B на численность (множество числовое, если все его элементы могут быть выражены целым или вещественным числом);

– проверка множеств на включение $A \subset B, B \subset A$ (все элементы множества A есть также и в B, и наоборот);

– проверка множеств A и B на пустоту и единичность (во множестве нет элементов, множество состоит из одного элемента);

– пересечение множеств $C = A \cup B$ (в множество C включить элементы, которые одновременно есть и в A, и в B);

– объединение множеств $C = A \cap B$ (в множество C включить элементы, которые одновременно есть или в A или в B);

– разность множеств $C = A \setminus B, C = B \setminus A$ (в множество C входят элементы множества A без элементов из множества B, и наоборот;

– симметрическая разность множеств $C = A \Delta B$ (в множество C входят элементы, которых одновременно нет в множестве A и B.

8. Редактор несимметричных деревьев

Редактор представляет собой приложение для осуществления базовых операций над несимметричным деревом (частный случай графа). Несимметричное дерево имеет корневой узел (вершину), с которой посредством ребер соединяются узлы последующего уровня и т.п. Узлы последнего уровня называются листьями. Число ребер на каждом уровне может быть разным, ребра могут иметь различный вес.

Визуальный интерфейс приложения состоит из области, в которой отображается дерево, блока ввода дерева и панели управления. Область отображения для дерева позволяет увидеть сам дерево и операции, выполняемые над ним. Узлы дерева отображаются в виде закрашенных окружностей, ребра обозначаются линиями. Все вершины должны быть подписаны (латинские буквы от A до Z), координаты вершин задаются константами, либо выбираются случайно. Корневая вершина обозначается буквой A. Веса ребер тоже должны отображаться. Блок ввода дерева представляет собой список из нескольких элементов, каждый элемент описывает одно ребро в виде двух букв (обозначения начальной и конечной вершин) и числа (вес ребра).

Панель управления представляет собой множество кнопок, по которым осуществляются базовые операции над деревом. Результат выполнения опера-

ции должен отображаться на дереве. Пока выполнение операции не завершится, остальные операции не должны быть доступны. Необходимо реализовать следующие операции:

– вычисление количества вершин, ребер, суммарного веса ребер;
– обход дерева в ширину (по шагам, каждый шаг по нажатию кнопки);
– обход дерева в глубину (по шагам, каждый шаг по нажатию кнопки);
– поиск кратчайшего пути (суммарный вес ребер) между заданными вершинами.

Также необходимо реализовать загрузку и сохранение дерева в файл.

9. Калькулятор систем линейных алгебраических уравнений (СЛАУ)

Калькулятор представляет собой приложение для решения СЛАУ и выполнения некоторых сопутствующих. Коэффициенты СЛАУ задаются в виде матрицы размерностью N<=10.

Визуально, интерфейс приложения представляет собой панель отображения СЛАУ в графическом режиме, блок ввода размерности матрицы и ее коэффициентов, панель результата, панель операций. Панель отображения СЛАУ предназначена для показа текущей системы уравнений. Отображение осуществляется как в математическом виде, так и в виде матрицы. Блок ввода исходных данных позволяет вводить коэффициенты СЛАУ. Для ввода коэффициента необходимо заполнить номер строки (номер уравнения), номер столбца (коэффициент уравнения), значение коэффициента, после чего путем нажатия кнопки «Добавить» коэффициент добавляется в систему, а область отображения перерисовывается. Панель результата предназначена для отображения результата вычислений в зависимости от выполняемой операции. Панель операций состоит из нескольких кнопок управления. Нажатие на каждую из кнопок приводит к выполнению некоторой операции. Необходимо реализовать следующие операции:

– вычисление определителя (для матриц размерностью более 3 определитель вычисляется через алгебраические дополнения, которые в свою очередь представляют определители меньшего порядка. Следовательно можно использовать рекурсию для организации вычислений);
– вычисление алгебраического дополнения для элемента матрицы I,J. Значения I,J задает пользователь в отдельном окне при запуске операции;
– решение СЛАУ методом Гаусса, позволяющий свести СЛАУ к треугольному виду (с обязательной пошаговой демонстрацией промежуточных этапов решения в панели отображения. Новый шаг по таймеру или после повторного нажатия кнопки запуска операции);
– решение СЛАУ методом Крамера, позволяющий получить решение путем вычисления определителя (с обязательной пошаговой демонстрацией промежуточных этапов решения в панели отображения. Новый шаг по таймеру или после повторного нажатия кнопки запуска операции);
– решение СЛАУ матричным методом, позволяющий получить решение путем вычисления обратной матрицы (с обязательной пошаговой демонстраци-

ей промежуточных этапов решения в панели отображения. Новый шаг по таймеру или после повторного нажатия кнопки запуска операции);

Также необходимо реализовать загрузку и сохранение СЛАУ в файл.

10. Клавиатурный тренажер

Тренажер представляет собой классическую программу для обучения методу слепой печати.

Визуально, интерфейс приложения представляет собой окно ввода, в котором отображается текст, который следует ввести. Текст, который еще не введен, отображается пониженной яркостью. Введенный текст отображается контрастным цветом (белым или черным в зависимости от фона). Под окном ввода располагается изображение клавиатуры, нажимаемые пользователем клавиши должны выделяться цветом. В режиме подсказки другим цветом также выделяется и следующий символ, который пользователю еще только предстоит ввести. Слева или справа находится панель управления, где располагаются основные кнопки управления «новая тренировка», «продолжить тренировку», «остановить тренировку», «настроить опции» (включить подсказку, цвет фона, цвета подсвечиваемых клавиш, указать максимальное количество допускаемых ошибок при выполнении упражнения).

Тренинг включает несколько упражнений. Упражнения хранятся в текстовом файле (или нескольких файлах). При нажатии на кнопку «новая тренировка» загружается первое упражнение. При нажатии на кнопку «продолжить тренировку» загружается текущее упражнение, на котором пользователь остановился. Номер этого упражнения необходимо сохранять в файле, как и статистику (время выполнения и аритмичность) всех пройденных упражнений.

При вводе первого символа в упражнении автоматически запускается таймер, работающий до окончания ввода всех символов упражнения. Если пользователь допустил ошибку, ввод символа не производится, приложение издает звук или кратковременно меняет цвет фона окна ввода. Также приложение должно подсчитывать аритмичность ввода, которая вычисляется как дисперсия значений времени, которое проходит между двумя нажатиями. По окончании упражнения приложение автоматически загружает следующее упражнение.

11. Визуализатор симметричного шифратора

Шифратор представляет собой приложение для кодирования и декодирования информации. Сегодня существует большое количество различного рода алгоритмов шифрации, как симметричных, так и ассиметричных. В симметричных шифрах для кодирования и декодирования информации используется один и тот же ключ. Ключ – это последовательность байт определенного размера. В приложении необходимо реализовать кодирование и декодирование двумя видами шифров: RC4 и «Поворотная решетка».

Визуально, интерфейс приложения состоит из поля данных и панели операций, в которой имеется несколько кнопок. При нажатии на кнопку «выбрать исходный файл» исходные данные считываются из файла (размер файла не более 256 байт, если больше приложение сообщает об ошибке) и помещаются в

поле данных (каждый символ отображать в 16-тиричном виде для удобства анализа в графическом режиме, сами же символы располагать в виде таблицы 16x16). При нажатии на кнопку «выбрать результирующий файл» пользователю выводится соответствующее сообщение. На панели также имеется переключатель выбора алгоритма. При нажатии на кнопку «ввести ключ» из файла считывается ключ. Если размер ключа не соответствует размеру ключа для данного алгоритма, выводится сообщение об ошибке.

После корректного ввода исходных данных и ключа становится доступной кнопка «кодировать/декодировать». Нажатие на данную кнопку запускает процедуру кодирования данных. При этом каждый шаг должен отображаться визуально в поле данных. Также в поле данных должна выводиться текстовая подсказка, что изменится на следующем шаге работы алгоритма. Очередной шаг работы алгоритма производится при нажатии на кнопку «кодировать/декодировать». Таким образом, чтобы закодировать или декодировать файл до конца, необходимо многократно нажимать на эту кнопку (данное приложение является визуализатором, призванным продемонстрировать работу алгоритма, а не шифрование в автоматическом режиме). На последнем шаге кодированная или декодированная информация записывается в результирующий файл. Также необходимо вести прогресс – показывать сколько шагов осталось до конца кодирования/декодирования. Описание работы алгоритмов RC4 и «Поворотная решетка» взять из открытых источников.

12. Визуализатор работы волнового алгоритма

Волновой алгоритм (алгоритм Lee) – один из самых интересных и широко используемых алгоритмов. Он позволяет находить кратчайший путь от точки А в точку Б в планарном графе, представляющий некоторый лабиринт. Этот алгоритм широко используется в игровой индустрии для построения кратчайшего пути к сопернику, что позволяет обходить препятствия на картах. Также алгоритм может быть использован для построения системы автоматического проектирования печатных плат в электронике (как известно, выводы электронных компонентов на плате соединяются медными дорожками, причем их длина выбирается кратчайшей). Наконец, с некоторыми модификациями, алгоритм может использоваться для моделирования реальных ситуаций, в том числе в трехмерных координатах – моделирование распространения тепла в металлической пластине или загрязняющей примеси в атмосферном воздухе. Описание алгоритма есть в открытых источниках.

Визуализатор работы волнового алгоритма представляет собой приложение, показывающее наглядно работу алгоритма. Интерфейс приложения включает большое поле, в котором пользователь может мышкой нарисовать произвольный лабиринт. Лабиринт представляется в виде матрицы, где стенки помечаются отрицательным числом, а проходимые места – нулями. Положительные числа могут быть использованы для моделирования распространения цифровой волны (первый этап работы алгоритма). Кроме того, в интерфейсе приложения есть панель настроек (где можно указать размеры лабиринта, координаты точек А и Б) и панель операций, где располагаются основные кнопки управления –

«загрузить карту лабиринта из файла», «сохранить карту лабиринта в файл», «начать моделирование», «следующий шаг», «очистить карту».

Первые две кнопки позволяют загрузить или сохранить нарисованную карту в файл. Последующая кнопка запускает визуализацию процесса построения пути. При нажатии кнопки «следующий шаг» на поле отображается текущий фронт волны в виде номера фронта волны (первая часть алгоритма), либо рисуется путь (вторая часть алгоритма).

13. Визуализатор приготовления блюда по заданному рецепту

Визуализатор приготовления блюда представляет собой приложение, которое создается в помощь начинающим кулинарам или просто домохозяйкам. Оно помогает визуально представить процесс приготовления блюда.

Рецепт описывается в виде текстового файла. Каждая строчка в этом файле состоит из названия этапа, времени, которое занимает этап и ссылки на изображение блюда на данном этапе. Приложение считывает файл построчно (в первой строке название рецепта) и отображает через указанные промежутки времени соответствующие изображения блюда. Вот как, например, может выглядеть рецепт приготовления яичницы:

```
ЯИЧНИЦА
Поставить сковороду на огонь 00:00 1.jpg
Налить 20 г. масла 00:05 2.jpg
Подождать 10 секунд 00:15 3.jpg
Разбить яйца 00:25 4.jpg
Подождать 40 секунд 01:05 5.jpg
Перевернуть яйца при помощи деревянной лопатки 01:25 6.jpg
Посолить 01:30 7.jpg
Подождать 40 секунд 02:15 8.jpg
Блюдо готово. Не забудьте выключить плиту. Приятного аппетита! 02:20 9.jpg
```

Изображения 1.jpg – 9.jpg необходимо подготовить заранее в графическом редакторе. Все они должны иметь размеры, соответствующие размеру области с видом блюда в визуализаторе. Так, например, изображение 1.jpg демонстрирует пустую сковороду (вид сверху), а изображение 7.jpg – перевернутую солонку над сковородой с почти готовым блюдом. Для демонстрации работы приложения необходимо составить не менее трех рецептов (в обычном блокноте) и подготовить соответствующие изображения.

Визуально приложение состоит из области, в которой демонстрируются этапы приготовления блюда (изображения), название этапа. Внизу области выводится таймер, который показывает текущее время (часы), время, когда приготовление блюда будет завершено, общее время приготовления блюда, сколько времени прошло с момента начала приготовления, и время, которое осталось до нового этапа. Кроме того, имеется панель управления, в которой располагаются кнопки «Загрузить рецепт», «Начать приготовление блюда» (запускает процесс приготовления), «Пауза» (кулинар может приостановить таймер, если видит, что очередной этап не завершен, например мясо в реальности не дожарилось), «Увеличить/уменьшить скорость» (данная кнопка сокращает или увеличивает пропорционально время всех этапов, используется, например тогда, когда из-за

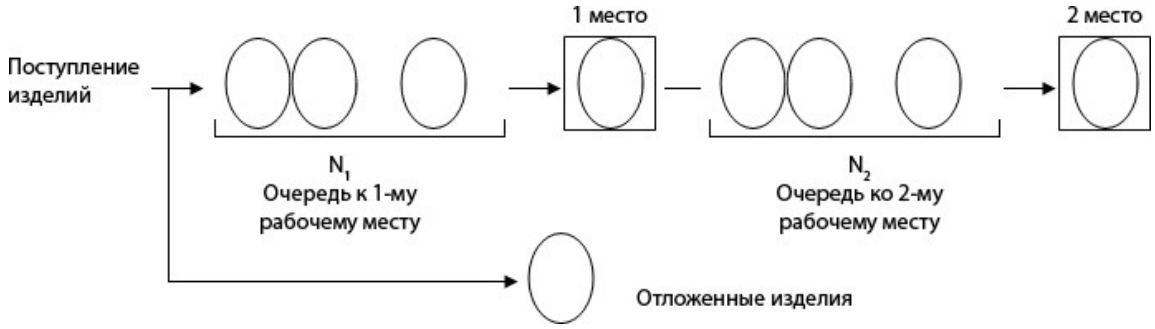
слабого газа время приготовления нужно увеличить), «пропустить этап». Окончание каждого этапа необходимо сопровождать звуковым сигналом (чтобы кулинар «не проспал», и не забыл вовремя посолить блюдо), однако эта опция может быть отключена в настройках. Также в настройках сохраняется и коэффициент скорости воспроизведения рецепта (по умолчанию 1).

Визуализатор также ведет статистику приготовления блюд. После окончания приготовления статистика обновляется. Статистика отображает: среднее время приготовления каждого блюда, количество приготовлений, популярное блюдо (критерий популярности можно придумать самостоятельно – например, максимальное количество приготовлений при минимальной стоимости ингредиентов, но тогда в рецепт придется добавить стоимость ингредиентов для каждого этапа). Предусмотреть возможность сброса статистики.

7.2. Моделирующие приложения

14. Моделирование работы производственной линии

На поточной линии предприятия выполняются две операции (соответственно два рабочих места), осуществляемые в строгой последовательности, т.е. вторая операция всегда следует за первой. Обрабатываемые изделия громоздки, поэтому на одной поточной линии могут находиться только N изделий, включая обрабатываемые. Между рабочими местами выделяется пространство, достаточное для размещения N_2 изделий, а перед первым рабочим местом – для N_1 изделий. Выполняется условие $N_1 + N_2 + 2 = N$. Обработка изделия, которое не может разместиться в пределах линии из-за недостатка свободного пространства, откладывается.



Интервалы времени между запросами на обработку изделий распределены равномерно в диапазоне $t_1...t_2$, времена обработки на первом и втором рабочем месте распределены в диапазоне $t_3...t_4$ и $t_5...t_6$ соответственно. Изделия автоматически транспортируются от одного рабочего места ко второму, временем транспортировки пренебречь. Если очередь ко второму рабочему месту заполнена до конца, т.е. в ней ожидают N_2 изделий, то происходит блокировка первого рабочего места, так как изделие с него не может быть убрано. На заблокированное рабочее место не может поступать для обработки другое изделие.

Визуально, приложение включает область, в которой наглядно отображается процесс моделирования в графическом режиме, область статистики, панель для ввода исходных параметров, кнопка «Старт/останов моделирования»

для выполнения шагов моделирования в автоматическом режиме по таймеру, кнопка «Следующий шаг» для ручного выполнения шагов моделирования.

В области статистики должна выводиться следующая информация:

– загрузка рабочих мест;
– среднее время обработки одного изделия на поточной линии;
– число изделий, обработка которых отложена;
– средняя длина очередей к первому и второму рабочему месту;
– доля времени, в течение которого первое место заблокировано;
– общее число обработанных изделий.

15. Моделирование работы танкерного флота

Флот, состоящий из N танкеров осуществляет перевозку сырой нефти из пункта А в пункт В. Предполагается, что все танкеры могут быть загружены в А одновременно. В пункте В имеется только один разгрузочный док, с которого разгружаемая нефть поступает в хранилище, а затем по трубопроводу – на очистительную установку. Нефть поступает в хранилище с разгружаемого в доке танкера с постоянной скоростью C_1 $м^3$/час. Хранилище непрерывно снабжает сырой нефтью очистительную установку с постоянной скоростью C_2 $м^3$/час. Разгрузочный док работает с 6:00 до 24:00. Правила безопасности требуют прекращения разгрузки в момент закрытия дока. Разгрузка танкера заканчивается, когда объём оставшейся в танкере нефти становится меньше K_1 $м^3$.

Ёмкость хранилища равна K_2 $м^3$. Когда хранилище заполнено до предела, разгрузка прерывается до тех пор, пока объём нефти в хранилище не снизится до 80% его ёмкости. Когда хранилище становится почти пустым (меньше K_3 $м^3$), снабжение очистительной установки прекращается до тех пор, пока объём нефти в хранилище не станет равным $10*K_3$ единиц. Это делается для устранения возможности частых остановок и запусков очистительной установки. Характеристики танкеров:

– номинальная грузоподъёмность – K_4 единиц;
– время в пути загруженного танкера распределено равномерно в диапазоне a_1...b_1 часов.
– время в пути пустого танкера распределено равномерно в диапазоне a_2...b_2 часов.
– время погрузки распределено равномерно в диапазоне a_3...b_3 часов.

Начальные условия моделирования соответствуют ситуации, когда хранилище заполнено на половину, а пустые танкеры находятся в пункте А в ожидании погрузки. Моделирование выполняется с фиксированным шагом в 1 час.

Визуально, приложение включает область, в которой наглядно отображается процесс моделирования в графическом режиме, область статистики, панель для ввода исходных параметров, кнопка «Старт/останов моделирования» для выполнения шагов моделирования в автоматическом режиме по таймеру, кнопка «Следующий шаг» для ручного выполнения шагов моделирования.

В области статистики должна выводиться следующая информация:

– время снабжения очистительной установки нефтью;
– средний объём нефти в хранилище;

– среднее время полного рейса танкера;
– среднее время ожидания танкера;
– среднее число танкеров, ожидающих разгрузки;
– общее количество перевезенной нефти.

16. Моделирование работы камерной печи

В процессе обработки на металлургическом заводе стальные отливки поступают в камерную печь с равномерно распределённым интервалом в диапазоне $a_1...b_1$. Отливки нагреваются в печи в целях дальнейшей рационализации хода технологического процесса. Прирост температуры отливки в печи за время 1 час (один шаг моделирования) описывается следующим уравнением:

$$\Delta h_i = (H - h_i)C,$$

где h_i – начальная температура *i*-ой отливки перед началом текущего шага моделирования; C – коэффициент скорости нагрева отливки, равномерно распределённая величина в диапазоне $a_2...b_2$; H – температура печи перед началом текущего шага моделирования.

Печь может раскаляться до максимальной температуры T с постоянным коэффициентом скорости нагрева, равным K, таким образом прирост температуры печи составляет:

$$\Delta H_+ = (T - H)K.$$

Отливки влияют друг на друга так, что помещение новой отливки в печь снижает температуру в печи и изменяет тем самым время нагрева уже находящихся в ней в данный момент отливок. Помещать новые отливки в печь можно при условии, что камера печи не заполнена до конца и только в начале каждого часа моделирования. Снижение температуры в печи при помещении новой отливки (или нескольких отливок) вычисляется по формуле

$$\Delta H_- = (H - h)NC,$$

где N – количество помещаемых отливок, h – начальная температура отливки (температура окружающей среды).

Таким образом, при помещении отливок в течение шага моделирования печь мгновенно остывает на величину ΔH_-, а затем разогревается на величину ΔH_+. Если отливки не помещать в печь, то в течение шага моделирования она просто разогревается на величину ΔH_+.

Максимальное количество отливок, которые можно загрузить в печь равно M. Если в тот момент, когда поступает новая отливка, печь не заполнена, то отливка помещается в печь. Если печь заполнена, отливка складируется рядом с печью.

Стратегия управления технологическим процессом состоит в том, что нагрев в печи отливок продолжается до тех пор, пока температура одной из них не достигнет $h_{\max}$. Как только это произойдет, все отливки выгружаются из печи. Если перед печью есть склад отливок, то они загружаются в печь и начинают греться.

Начальная температура печи равна $H_{\min}$.

Визуально, приложение включает область, в которой наглядно отображается процесс моделирования в графическом режиме, область статистики, панель для ввода исходных параметров, кнопка «Старт/останов моделирования» для выполнения шагов моделирования в автоматическом режиме по таймеру, кнопка «Следующий шаг» для ручного выполнения шагов моделирования.

В области статистики должна выводиться следующая информация:

– среднее время нагрева отливок;
– среднее время ожидания холодных отливок перед печью;
– загрузка камерной печи;
– количество отливок, которые ожидают перед печью в данный момент;
– количество поступивших отливок;
– количество обработанных отливок;
– количество отливок, находящихся в печи.

17. Моделирование процесса лечения в стационаре

Количество пациентов, поступающих в стационар за день, равномерно распределено в диапазоне $a_1...b_1$. Каждый пациент проходит обследование при поступлении и в ходе лечения раз в сутки. Результат обследования отражает условный коэффициент состояния пациента. Для поступающих на лечение пациентов этот коэффициент равномерно распределен в диапазоне от $a_2...b_2$. Когда в стационаре нет свободных мест, пациенты с коэффициентом выше a_3 не принимаются на лечение, а направляются в местную поликлинику. Если этот коэффициент ниже a_3, то стационар может досрочно выписать другого пациента, у которого коэффициент состояния стал выше a_3 в ходе лечения и положить этого пациента на его место. Если и выписать досрочно никого нельзя, стационар направляет поступившего «тяжелого» пациента в местную поликлинику, несмотря на его плохое состояние.

Пациенты с коэффициентом состояния выше a_3 и «тяжелые» пациенты отправляются на лечение (долечивание) в местную поликлинику – тогда и только тогда, когда в стационаре нет свободных мест. Но даже если в будущем место в стационаре появилось, направленный в поликлинику пациент обратно в стационар не возвращается. Если свободных мест в местной поликлинике также нет, больному отказывают в лечении.

В течение суток коэффициент состояния пациента при лечении в стационаре увеличивается на величину, равномерно распределённую в диапазоне $a_4...b_4$, а при лечении в местной поликлинике на величину равномерно распределённую в диапазоне $a_5...b_5$ (меньше $a_4...b_4$). Всего в стационаре N мест, а местная поликлиника способна обслужить в сутки К пациентов. Пациент выписывается из стационара или местной поликлиники, когда результат обследования становится выше a_6 баллов. Первоначально стационар и местная поликлиника пусты. Интервал моделирования составляет ровно одни сутки.

Визуально, приложение включает область, в которой наглядно отображается процесс моделирования в графическом режиме, область статистики, панель для ввода исходных параметров, кнопка «Старт/останов моделирования»

для выполнения шагов моделирования в автоматическом режиме по таймеру, кнопка «Следующий шаг» для ручного выполнения шагов моделирования.

В области статистики должна выводиться следующая информация:

– среднее время пребывания пациентов в стационаре;

– среднее время лечения пациентов в местной поликлинике;

– количество направленных в местную поликлинику для долечивания;

– количество направленных в местную поликлинику «тяжелых» пациентов;

– количество пациентов, которым отказали в лечении и стационар, и местная поликлиника;

– количество поступивших и вылечившихся пациентов;

– количество пациентов, находящихся в данный момент на лечении в стационаре и местной поликлинике.

18. Моделирование работы банка для автомобилистов

В банке для автомобилистов имеется два окошка, каждое из которых обслуживается одним кассиром и имеет отдельную подъездную полосу. Обе полосы расположены рядом. Из предыдущих наблюдений известно, что клиенты прибывают через интервалы, равномерно распределённые в диапазоне $a_1...b_1$ минут. П0родолжительность обслуживания обоих кассиров одинакова и равномерно распределена в диапазоне $a_2...b_2$ минут. Известно также, что при равной длине очередей, а также при отсутствии очередей, клиенты отдают предпочтение первой полосе. Во всех других случаях клиенты выбирают более короткую очередь. После того, как клиент въехал в банк, он не может покинуть его, пока не будет обслужен. Однако он может сменить очередь, если стоит последним и разница в длине очередей при этом составляет не менее двух автомобилей. На каждой полосе может находиться не более N автомобилей. В каждой очереди может находиться не более K автомобилей. Если место перед банком заполнено до отказа (обе очереди заполнены по максимуму), прибывший клиент уезжает.

Изначально оба кассира свободны и обе очереди пусты. Шаг моделирования составляет 1 минуту.

Визуально, приложение включает область, в которой наглядно отображается процесс моделирования в графическом режиме, область статистики, панель для ввода исходных параметров, кнопка «Старт/останов моделирования» для выполнения шагов моделирования в автоматическом режиме по таймеру, кнопка «Следующий шаг» для ручного выполнения шагов моделирования.

В области статистики должна выводиться следующая информация:

– количество клиентов, обслуженных первым и вторым кассирами;

– общее количество обслуженных клиентов;

– среднее время пребывания клиента в банке;

– среднее время пребывания клиента в очереди;

– процент и количество клиентов, которые отказались от обслуживания;

– количество смен очередей клиентами;

– загруженность кассиров (отношение времени работы кассира к общему времени моделирования;

19. Моделирование работы морского порта

В морском порту нефть загружается в танкеры, которую затем доставляют покупателю на нефтебазу. Порт может одновременно загружать не более K танкеров одновременно. Танкеры, прибывающие в порт с интервалом, равномерно распределенном в диапазоне a_1...b_1 часов, относятся к трем различным типам. Значения относительной частоты появления танкеров данного типа и времени, требуемого на их погрузку, приведены ниже:

Тип	Относительная частота	Время погрузки, ч	Вместимость, тыс.тонн
1	0.25	a_2...b_2	v_2
2	0.55	a_3...b_3	v_3
3	0.20	a_4...b_4	v_4

Очередной танкер, прибывающий в порт, сразу становится на погрузку, если порт может его обслужить, иначе ожидает в очереди. После погрузки танкер (любого типа) сразу отчаливает и находится в пути ровно a_5 часов, достигая за это время пункта назначения, где и происходит разгрузка. Разгрузка танкера занимает столько же времени, сколько и погрузка. После этого танкер возвращается обратно также в течение a_5 часов.

Процесс перевозки нефти осложняют штормы, воздействию которых подвергается как порт, так и танкеры. Штормы возникают с интервалом, равномерно распределенном в диапазоне a_5...b_5 часов и продолжаются a_6...b_6 часов. Во время шторма порт закрывается и загрузка приостанавливается, а все танкеры замедляют скорость своего передвижения, из-за чего время в пути увеличивается на величину из диапазона a_7...b_7 часов. Время одного шага моделирования составляет 1 час.

Покупателю необходимо иметь M тыс. тонн нефти ежесуточно, иначе производство понесет убытки.

Визуально, приложение включает область, в которой наглядно отображается процесс моделирования в графическом режиме, область статистики, панель для ввода исходных параметров, кнопка «Старт/останов моделирования» для выполнения шагов моделирования в автоматическом режиме по таймеру, кнопка «Следующий шаг» для ручного выполнения шагов моделирования.

В области статистики должна выводиться следующая информация:

– количество погрузок и разгрузок танкеров (каждого из трех типов);

– количество доступной нефти на нефтебазе;

– среднее время танкеров в пути с учетом осложнений (каждого из трех типов);

– средняя длина очереди на погрузку;

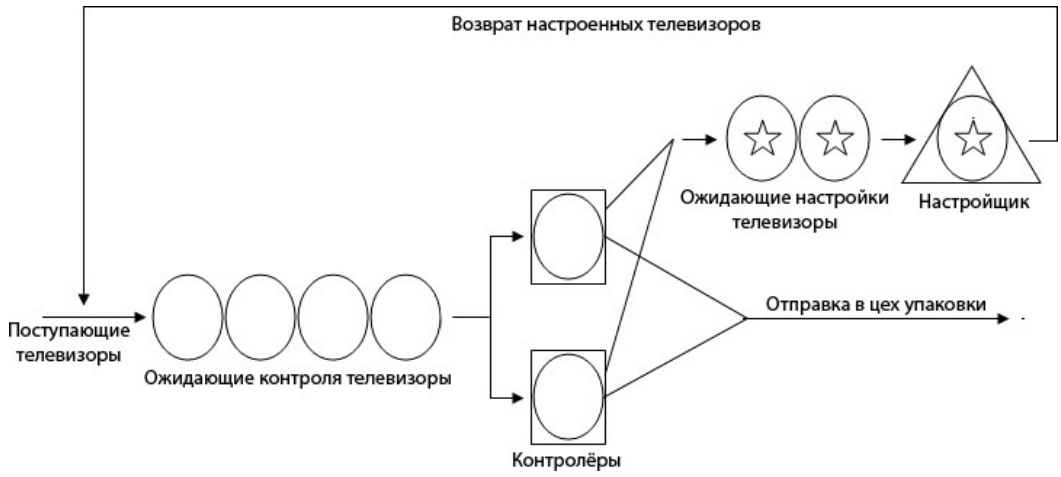
– количество нефти, доставленной танкерами в течение суток (обновляется 1 раз в 24 часа);

– среднее количество танкеров, попавших под шторм (вычисляется как среднее арифметическое отношения количества танкеров, попавших под шторм, к общему числу танкеров).

20. Моделирование работы производственной поточной линии с пунктами технического контроля

Собранные телевизоры на заключительной стадии их производства проходят ряд пунктов технического контроля. В последнем из этих пунктов осуществляется проверка настройки телевизоров. Если при проверке обнаружилось, что телевизор работает некачественно, он направляется в пункт настройки, где настраивается заново. После перенастройки телевизор снова направляется в последний пункт контроля для проверки качества настройки. Телевизоры, которые сразу или после нескольких возвратов в пункт настройки прошли фазу заключительной проверки, направляются в цех упаковки.

На схеме кружками пустыми кружками обозначены телевизоры, которые ожидают проверки, кружками со звездочкой – телевизоры, которые стоят в очереди к пункту настройки, либо настраиваются. Время между поступлениями новых телевизоров в пункт контроля распределено равномерно на интервале a_1...b_1 мин. В пункте заключительной проверки параллельно работают два контролера. Время, необходимое на проверку одного телевизора, распределено равномерно на интервале a_2...b_2 мин. В среднем N% телевизоров проходят проверку и направляются на упаковку. Остальные телевизоры отправляются в пункт настройки, где работает один настройщик. Время настройки распределено равномерно на интервале a_3...b_3 мин. Если настройщик или контроллеры заняты, то поступающие телевизоры образуют очереди, размеры которых неограниченны. Очередь к контроллерам общая. Шаг моделирования – 1 минута.



Визуально, приложение включает область, в которой наглядно отображается процесс моделирования в графическом режиме, область статистики, панель для ввода исходных параметров, кнопка «Старт/останов моделирования» для выполнения шагов моделирования в автоматическом режиме по таймеру, кнопка «Следующий шаг» для ручного выполнения шагов моделирования.

В области статистики должна выводиться следующая информация:

– число телевизоров в очереди к контроллерам;

– число телевизоров в очереди к настройщику;

– средняя загруженность первого и второго контроллера (вычисляется как время работы контроллера, деленная на общее время моделирования);

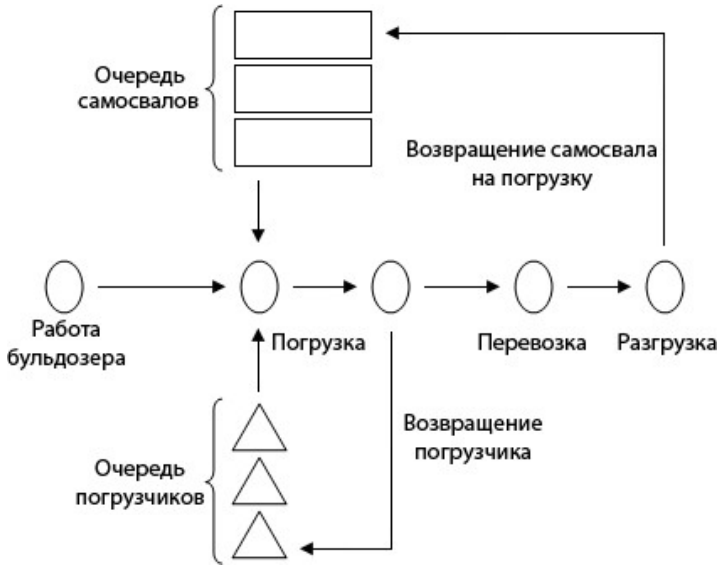
– средняя загруженность настройщика;

– количество отбракованных телевизоров, попавших к настройщику;
– количество настроенных телевизоров;
– количество поступивших телевизоров;
– количество готовых к упаковке телевизоров;
– среднее время, которое тратится на обслуживание одного телевизора (с учетом ожидания в очередях, возможной настройки).

21. Моделирование автогрузовых перевозок

Моделируемая система состоит из одного бульдозера, n_1 самосвалов и n_2 механизированных погрузчиков. Бульдозер сгребает землю к погрузчикам. Время, затрачиваемое бульдозером на формирование одной кучи земли, имеет равномерное распределение в интервале $a_1...b_1$. Одна куча земли в точности помещается в один самосвал. Кроме наличия земли для начала погрузки требуются погрузчик и пустой самосвал. Время погрузки распределено равномерно и составляет $a_2...b_2$ для i-того погрузчика.

После того как самосвал загружен, он уезжает к месту разгрузки, разгружается и вновь возвращается на погрузку. Время нахождения самосвала в пути распределено равномерно, причем в загруженном состоянии он тратит на дорогу $a_3...b_3$ мин, а в незагруженном – $a_4...b_4$ мин. Время разгрузки распределено равномерно на интервале $a_5...b_5$ мин. После погрузки погрузчик должен отдыхать в течение a_6 мин., а самосвал – в течение a_7 мин., однако они могут делать это, находясь в очереди погрузчиков или очереди самосвалов соответственно. Очереди могут образоваться в частности из-за медленной работы бульдозера. Если погрузчик (или самосвал) как следует не отдохнул (время a_6 или a_7 не закончилось, но очередь на погрузку или перевозку подошла), он уступает право погрузки или перевозки стоящему за ним погрузчику (или самосвалу). Шаг моделирования – 1 минута.



Визуально, приложение включает область, в которой наглядно отображается процесс моделирования в графическом режиме, область статистики, панель для ввода исходных параметров, кнопка «Старт/останов моделирования»

для выполнения шагов моделирования в автоматическом режиме по таймеру, кнопка «Следующий шаг» для ручного выполнения шагов моделирования.

В области статистики должна выводиться следующая информация:

– число самосвалов в очереди;

– число погрузчиков в очереди;

– число самосвалов в пути;

– средняя загруженность самосвалов и погрузчиков (вычисляется как время работы (не отдыха, и не ожидания в очереди) самосвала или погрузчика, деленная на общее время моделирования);

– общее количество перевезенных куч земли

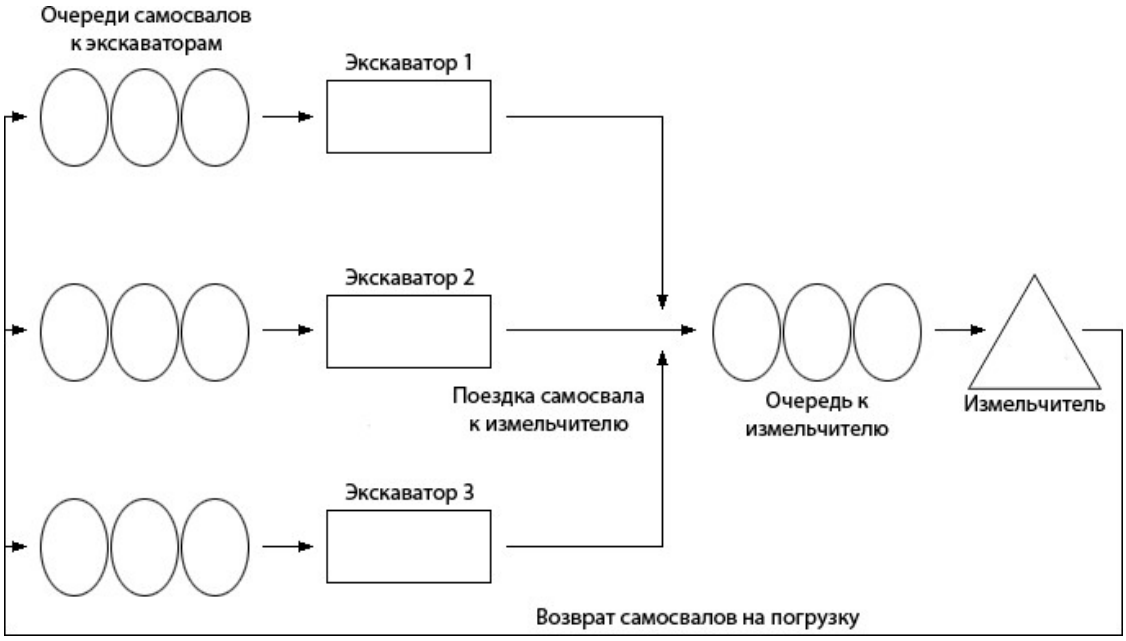
– количество куч земли, доступных к погрузке;

– среднее время, требуемое для погрузки, перевозки и разгрузки одной кучи (с учетом простоя техники).

22. Моделирование работы карьера

В карьере самосвалы доставляют руду от трех экскаваторов, причем после выгрузки руды у измельчителя самосвалы всегда возвращаются к одним и тем же закрепленным за ними экскаваторам. Используются самосвалы двух типов грузоподъемностью 20 и 50 тонн. От грузоподъемности самосвала зависят его параметры: время погрузки, поездки до измельчителя, разгрузки и обратной поездки к своему экскаватору.

Тип	Время погрузки, мин	Время поездки до измельчителя, мин	Время разгрузки, мин	Время поездки к экскаватору, мин
20 тонн	$a_2...b_2$	$a_1...b_1$	$a_3...b_3$	$a_4...b_4$
50 тонн	$a_6...b_6$	$a_5...b_5$	$a_7...b_7$	$a_8...b_8$



К каждому экскаватору (i=1..3) приписаны n_i 20-титонных и k_i 50-титонных самосвалов. Из-за длительного времени погрузки и разгрузки у экс-

каваторов и измельчителя могут образоваться очереди из самосвалов, причем очередь у измельчителя общая. Шаг моделирования – 1 минута.

Визуально, приложение включает область, в которой наглядно отображается процесс моделирования в графическом режиме, область статистики, панель для ввода исходных параметров, кнопка «Старт/останов моделирования» для выполнения шагов моделирования в автоматическом режиме по таймеру, кнопка «Следующий шаг» для ручного выполнения шагов моделирования.

В области статистики должна выводиться следующая информация:

– число самосвалов в каждой из трех очередей к экскаваторам и в очереди к измельчителю;

– число самосвалов на погрузке в данный момент (1...3);

– число самосвалов в пути;

– средняя загруженность самосвалов (вычисляется как время работы (движение, погрузка, разгрузка) самосвалов, деленная на общее время моделирования);

– средняя загруженность экскаваторов (вычисляется как время работы экскаватора, деленная на общее время моделирования);

– общее количество руды, доставленное к измельчителю (в тоннах).

23. Моделирование потока транспорта на участке дороги с односторонним движением

Моделируемая система представляет собой поток транспорта в двух направлениях по дороге с двусторонним движением, одна сторона которой закрыта в связи с ремонтом. Длина ремонтируемого участка N метров. На обоих концах ремонтируемого участка решено было разместить два светофора для управления движением. Светофоры работают в автоматическом режиме по циклической схеме из четырех состояний: «левый зеленый, правый красный» (движение слева направо), «оба красных» (движение запрещено, однако заехавший на ремонтируемый участок может продолжать движение), «левый красный, правый зеленый» (движение справа налево), «оба красных». Состояние «оба красных» необходимо для того, чтобы заехавший на ремонтируемый участок дороги транспорт смог покинуть его, прежде чем откроется движение по этой же дороге в обратном направлении). Переключение светофоров из состояния в состояние осуществляется через каждые K_1, K_2, K_3, K_4 минут соответственно.

По дороге могут двигаться автомобили двух типов: легковые и грузовые. Легковые автомобили подъезжают к дороге слева через интервалы времени, равномерно распределенные в интервале $a_1...b_1$ минут, справа – в интервале $a_2...b_2$ минут, грузовые автомобили подъезжают к дороге слева через интервалы времени, равномерно распределенные в интервале $a_3...b_3$ минут, справа – в интервале $a_4...b_4$ минут, средняя скорость движения легковых и грузовых автомобилей (в обоих направлениях) равномерно распределена в интервалах $a_5...b_5$ и $a_6...b_6$ метров/мин соответственно.

При настройках светофора, не соответствующих текущей ситуации на дороге, либо при большой интенсивности движения транспорта возможны очереди как слева, так и справа ремонтируемого участка. Кроме того, возможна

ситуация, при которой транспорт, движущийся в одном направлении, не успел покинуть ремонтируемый участок (состояние «оба красных» слишком мало), в то время как навстречу ему уже устремился поток автомобилей. Такая ситуация именуется «пробкой». Для разрешения этой ситуации вызывают специальный вертолет-эвакуатор, который эвакуирует все машины, оказавшиеся на участке ремонтируемой дороги по воздуху, после чего движение продолжается. Вертолет прибывает мгновенно, погрузка машин и их эвакуация занимает a_7 минут, независимо от их количества. Пока продолжается эвакуация, машины двигаться по ремонтируемому участку не могут, но могут прибывать и слева, и справа. Шаг моделирования – 1 минута.

Визуально, приложение включает область, в которой наглядно отображается процесс моделирования в графическом режиме, область статистики, панель для ввода исходных параметров, кнопка «Старт/останов моделирования» для выполнения шагов моделирования в автоматическом режиме по таймеру, кнопка «Следующий шаг» для ручного выполнения шагов моделирования.

В области статистики должна выводиться следующая информация:

– среднее число легковых машин в очереди слева и справа;

– среднее число грузовых автомобилей в очереди слева и справа;

– среднее время прохождения транспорта через ремонтируемый участок дороги слева и справа, включая ожидание в очереди;

– число легковых машин, успешно проехавших через ремонтируемый участок дороги слева и справа;

– число грузовых автомобилей, успешно проехавших через ремонтируемый участок дороги слева и справа;

– число легковых машин и грузовых автомобилей, которые потребовалось эвакуировать для восстановления движения.

24. Моделирование планирования работ в цехе

В цехе имеется шесть станков, осуществляющих разные операции. По предварительной оценке время выполнения работы на каждом станке распределено равномерно в диапазоне $a_1...b_1$ мин.

Изделия трех типов для обработки поступают в цех с интервалом, равномерно распределенным в диапазоне $a_2...b_2$ мин. Для обработки изделия первого типа требуется K_1 операций, для обработки изделия второго типа – K_2 операций, для третьего – K_3 операций. Каждая операция осуществляется на соответствующем станке, поэтому K_i не могут быть больше шести. Какие операции необходимо выполнить, указывается в сопроводительном бланке на об-

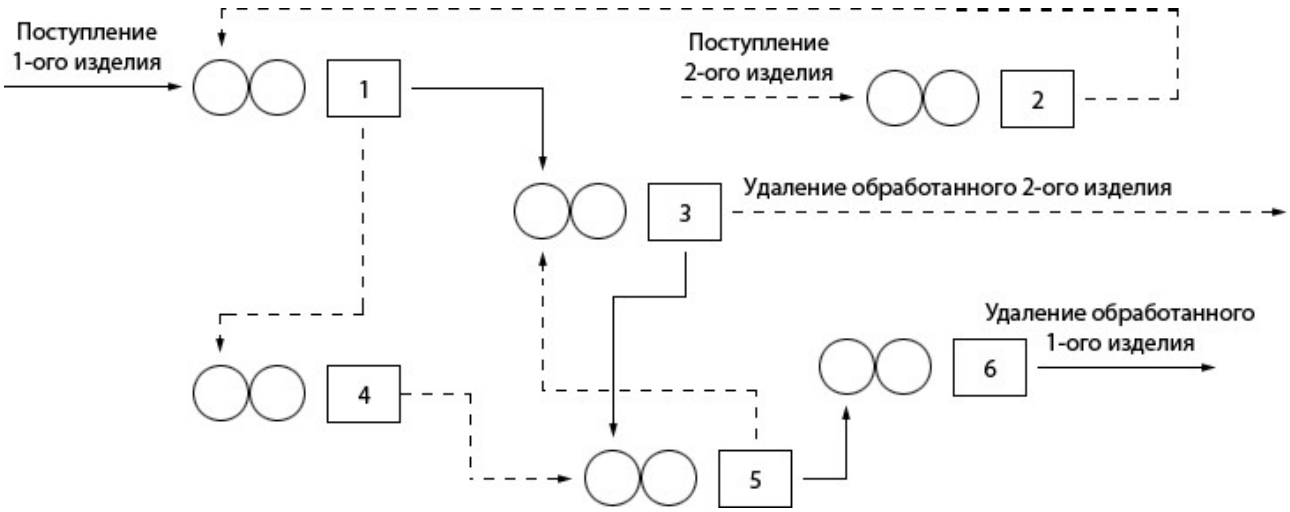
работку изделия. Перечень выполняемых операций задается в виде списка из чисел от 1 до 6 для каждого из типов изделий.

Из-за большого количества операций определенного типа возле некоторых станков может образовываться очередь из изделий. Но следует отметить, что порядок операций, выполняемых на станках, может быть изменен (например, сверловка отверстий может быть выполнена либо после фрезеровки, либо после сгибания детали). Поэтому для изделия каждый раз выбирается та операция, где очередь минимальна. Операции по обработке изделия не могут повторяться, также необходимо выполнять только те операции, которые указаны в сопроводительном бланке.

Перечень взаимозаменяемых операций представлен в таблице.

Номер операции	Операции, которые могут быть выполнены вместо этой операции
1	2, 3, 6
2	4, 5, 6
3	1, 2, 4, 5
4	1, 2, 6
5	3, 5, 6
6	1, 3, 4

Время перемещения изделия между станками постоянно и составляет a_3 мин. независимо от вида операции и соответствующего станка. Станки в цеху расположены так, как показано на рисунке. Шаг моделирования – 1 минута.



Визуально, приложение включает область, в которой наглядно отображается процесс моделирования в графическом режиме, область статистики, панель для ввода исходных параметров, кнопка «Старт/останов моделирования» для выполнения шагов моделирования в автоматическом режиме по таймеру, кнопка «Следующий шаг» для ручного выполнения шагов моделирования.

В области статистики должна выводиться следующая информация:

– общее количество обработанных изделий;
– общее количество изделий, поступивших на обработку;

– среднее количество изделий, находящихся в очереди к каждому из станков;

– загруженность каждого из 6-ти станков (вычисляется как отношение времени работы станка к общему времени моделирования);

– общее количество изменений в порядке выполнения операций;

– среднее время обработки одного изделия (с учетом ожидания в очередях) для каждого из трех типов.

25. Моделирование обучения магов школы Камартадж

В Камартадж поступают новые ученики, их количество равномерно распределено в диапазоне $a_1...b_1$. Поступление происходит раз в месяц, причем поступить может не более K учеников. Если в Камартадже уже обучается N учеников, новые ученики не могут поступить, пока старые не закончат обучение. В этом случае образуется общая очередь из поступающих.

Обучение включает в себя овладение тремя дисциплинами: базовые манипуляции пространством (БМП), боевая магия (БМ), высшая магия и использование артефактов (ВМИА). Обучение каждой дисциплине начинается только после полного овладения предыдущей, и только в указанном порядке. Время обучения каждой дисциплине составляет $a_2...b_2$, $a_3...b_3$, $a_4...b_4$ дней соответственно. Для определения того, овладел ли маг определённой дисциплиной, он проходит экзамен. на котором выставляется оценка по 100-бальной шкале. Если маг набирает k_2, k_3, k_4 баллов и выше по соответствующим дисциплинам, то считается, что он овладел дисциплиной и может переходить к изучению следующей. Если нет, то мага направляют на повторное изучение дисциплины, которое снова длится $a_2...b_2$, $a_3...b_3$, $a_4...b_4$ дней. После этого он обязан сдать экзамен повторно, но проходной балл составляет уже k_2+step, k_3+step, k_4+step баллов. Если экзамен снова не сдан, мага опять отправляют на повторное изучение дисциплины. Для третьей, последней попытки сдачи экзамена проходной балл составляет k_2+2*step, k_3+2*step, k_4+2*step баллов. Если экзамен не сдан и с третьего раза, мага отчисляют как непригодного. После успешной сдачи всех трех экзаменов обучение мага считается законченным, и он покидает Камартадж. Экзамены проводятся раз в неделю (каждый седьмой день с момента начала моделирования). Шаг моделирования составляет 1 день.

Визуально, приложение включает область, в которой наглядно отображается процесс моделирования в графическом режиме, область статистики, панель для ввода исходных параметров, кнопка «Старт/останов моделирования» для выполнения шагов моделирования в автоматическом режиме по таймеру, кнопка «Следующий шаг» для ручного выполнения шагов моделирования.

В области статистики должна выводиться следующая информация:

– количество поступивших на обучение магов;

– количество ожидающих обучения магов;

– количество завершивших обучение магов;

– количество отчисленных претендентов;

– количество повторов изучения каждой дисциплины;

– среднее время обучения (с учетом повторных изучений дисциплин).

26. Моделирование цикла жизни Звезды Смерти

Звезда Смерти (ЗС) – это астрологическое сооружение из вселенной «Звёздных войн», которая оснащена сверхмощным энергетическим лазерным оружием чрезвычайно разрушительной силы, способным уничтожать целые планеты.

Строительство Звезды Смерти происходит в 10 этапов. Каждый этап занимает определённое количество времени в интервале, равномерно распределенном в диапазоне a_1*k...b_1*k суток, где k – это номер этапа. Для строительства очередного этапа необходимо E_c*k МДж энергии в сутки и M_c*k тонн материалов в сутки. Энергия образуется за счет солнечных элементов, материалы же подвозят транспортные корабли. Солнечные элементы способны вырабатывать E_m МДж в сутки, вырабатываемая ими энергия при необходимости сохраняется на следующие сутки в специальных аккумуляторах. Транспортные корабли прибывают с интервалом, равномерно распределенным в диапазоне a_2...b_2 суток, разгружаются на складе в течение a_3...b_3 суток и убывают обратно. Одновременно может разгружаться только один транспортный корабль, поэтому остальные вынуждены ожидать в очереди. Один транспортный корабль способен привезти M_m тонн материалов. Если для строительства не хватает материалов или энергии, то строительство ЗС приостанавливается на целые сутки.

Повстанцам, разумеется, известно о строительстве, и они нападают на постройку с интервалом, равномерно распределенным в диапазоне a_4...b_4 суток и каждый раз наносят урон ЗС на величину, равномерно распределенную в диапазоне a_5...b_5 % прочности. При этом процесс строительства приходится приостанавливать и ремонтировать повреждения, так как для ремонта повреждений используются те же роботы, что и для строительства. Во время атаки, ни ремонт, ни строительство производиться не могут. Роботы могут восстановить a_6...b_6 % прочности в сутки, использовав для этого дополнительные E_c МДж энергии и M_c тонн материалов. Солнечные элементы и транспортные корабли повстанцы не атакуют. Считается, что неповрежденная ЗС имеет 100% прочности.

Если удастся достроить ЗС при таких условиях, то она немедленно будет приведена в действие, после чего погибнут все близлежащие планеты. Если достроить не получится (прочность отстроенного упадет до 0%, т.е. повстанцы разрушат все, что было построено на предыдущих этапах), Дарт Вейдер поймет, что ЗС не может быть построена в этой части галактики. И в том, и в другом случае моделирование останавливается. Шаг моделирования составляет 1 сутки.

Визуально, приложение включает область, в которой наглядно отображается процесс моделирования в графическом режиме, область статистики, панель для ввода исходных параметров, кнопка «Старт/останов моделирования» для выполнения шагов моделирования в автоматическом режиме по таймеру, кнопка «Следующий шаг» для ручного выполнения шагов моделирования.

В области статистики должна выводиться следующая информация:

– статус в данный момент (строительство, ремонт, атака);
– количество отстроенных этапов в данный момент;

– количество потраченной на строительство энергии и количество потраченных материалов;

– количество потраченной на ремонт энергии и количество потраченных материалов;

– количество суток, когда строительство или ремонт были остановлены по причине нехватки ресурсов;

– среднее число транспортных кораблей в очереди на разгрузку;

– среднее время, в течение которого наблюдались налеты повстанцев (вычисляется как отношение числа суток со статусом атака к общему времени моделирования);

– если моделирование завершилось – общее число суток которые прошли и результат моделирования (достроена ЗС или нет).

27. Моделирование обслуживания посетителей банка

В банке имеется N кассиров, M операторов (N, M<5) и один охранник на входе. Кассиры осуществляют только финансовые операции, операторы же помогают клиентам открывать счета, оформляют пластиковые карты, оформляют переводы, осуществляют консультирование и т.п.

Посетители приходят в банк с интервалом, равномерно распределенным в диапазоне $a_1...b_1$ минут. На входе их встречает охранник, который провожает посетителя либо в кассу, либо к операторам, либо, и туда, и туда сразу, в зависимости от типа услуг, которые требуется посетителю. На консультацию у охранника требуется a_2 минут.

Каждый посетитель может прийти в банк для оказания нескольких услуг сразу, но не более K. Среди этих услуг могут встречаться только финансовые (когда требуется посетить только кассира, например снять деньги), только банковские (когда требуется посетить только оператора, например оформить карту) и смешанные (когда требуется посетить сначала оператора, а затем кассира, например, открыть счет и положить на него деньги). Среди всех предоставляемых банком услуг a_3% являются финансовыми, a_4% являются банковскими, остальные смешанные.

Оператор и кассир тратят на оказание одной услуги время, равномерно распределенное в диапазоне $a_5...b_5$ минут и $a_6...b_6$ минут соответственно, причем это завит от вида услуги. Если к примеру, посетитель пришел для оказания двух смешанных и одной финансовой услуги, оператору потребуется $2*a_5...2*b_5$ времени, а кассиру $3*a_6...3*b_6$, а полное время обслуживания посетителя составит соответственно $2*a_5...2*b_5 + 3*a_6...3*b_6$ минут.

По совету охранника в зависимости от требуемых услуг посетитель занимает очередь в одну из касс, к одному из операторов, или к обоим сразу. При этом из соображений экономии своего времени посетитель выбирает самые короткие очереди. Если первой подходит очередь на обслуживание к оператору, посетитель обслуживается у него, а потом возвращается в очередь к кассиру. Если очередь в кассу подошла в тот момент, когда посетитель обслуживается оператором, он вынужден пропустить ее, и, после обслуживания оператором, занять очередь заново. Если первой подходит очередь в кассу, посетитель мо-

жет быть обслужен по такому же принципу, но только если у него нет смешанных услуг (следует помнить, что в смешанных услугах посетитель должен быть обслужен сначала оператором, а потом кассиром, но не наоборот). Шаг - моделирования составляет 1 минуту.

Визуально, приложение включает область, в которой наглядно отображается процесс моделирования в графическом режиме, область статистики, панель для ввода исходных параметров, кнопка «Старт/останов моделирования» для выполнения шагов моделирования в автоматическом режиме по таймеру, кнопка «Следующий шаг» для ручного выполнения шагов моделирования.

В области статистики должна выводиться следующая информация:

– количество пришедших посетителей;

– количество обслуженных посетителей;

– количество предоставленных услуг (финансовых, банковских, смешанных);

– средняя загруженность кассиров и операторов (определяется для каждого кассира в отдельности как отношение времени его работы к общему времени моделирования;

– средняя время обслуживания одного посетителя;

– количество ситуаций, когда пользователь вынужден был пропустить свою очередь.

28. Моделирование совершения налетов бандой разбойников

Банда из K разбойников совершает налеты на государственный банк с интервалом, равномерно распределенным в диапазоне a_1...b_1 суток. Полицейские мгновенно приезжают на место преступления и ликвидируют a_2% численного состава банды, еще a_3% арестовывают, остальным же удается уйти с золотом на сумму S $. P % этой суммы делится поровну между выжившими и не арестованными участниками банды, остальное уходит в «общак».

Для восполнения потерь банда занимается вербовкой. Известно, что вербовка нового члена банды может быть выполнена в интервал времени, равномерно распределенный в диапазоне a_4...b_4 суток. Для вербовки нового члена банды требуется выплатить ему M $. Арестованные члены банды также могут быть выкуплены за L $, но успешность этой операции составляет a_5%. Если операция оказалась успешной, то арестованный возвращается в банду, ему ничего не выплачивается. Если операция оказалась неуспешной, то члена банды, которого отправили переговорщиком к полицейским, арестовывают, а врученные ему L $ для выкупа, снова возвращаются в государственный банк. Всего в день можно совершить не более одной попытки выкупа. Для вербовки и выкупа деньги берутся из «общака», причем выкуп имеет приоритет. Если денег в «общаке» нет, то вербовка и выкуп не производятся, а банда в следующий раз «идет на дело» в усеченном составе, но при условии, что число членов банды не стало меньше K_{min}. Если же это число станет меньше K_{min}, то моделирование останавливается (считается, что банда расформирована). Банда не «идет на дело», пока есть возможность выкупить старых и навербовать новых членов банды, но не более K человек.

Количество денег в государственном банке изначально составляет S_{max} $ (S_{max}>S), и пополняется каждые сутки на величину, равномерно распределенную в диапазоне a_6...b_6. Кроме того, как уже известно, часть украденного может также быть возвращена в банк. Если в банке осталось менее S $ денег, то банда не совершает налета на такой банк, а ждет, пока количество средств в банке не восполнится. Опасаясь кражи, каждые a_7...b_7 суток дирекция банка переправляет U% всех денег в подземное хранилище, куда бандитам нет доступа.

Визуально, приложение включает область, в которой наглядно отображается процесс моделирования в графическом режиме, область статистики, панель для ввода исходных параметров, кнопка «Старт/останов моделирования» для выполнения шагов моделирования в автоматическом режиме по таймеру, кнопка «Следующий шаг» для ручного выполнения шагов моделирования.

В области статистики должна выводиться следующая информация:

– количество совершенных налетов;

– количество украденных денег;

– количество денег в банке

– количество денег, переправленных в подземное хранилище;

– количество выкупленных членов банды;

– количество неудачных попыток выкупа и количество возвращенных в банк денег;

– количество завербованных новых членов и количество потраченных на это денег;

– средняя сумма в «общаке» (вычисляется как среднее арифметическое денег в общаке каждые сутки за все время моделирования).

29. Моделирование загруженности перекрестка

Перекресток представляет собой пересечение главной и второстепенной дорог. Главная дорога состоит из двух полос, второстепенная – однополосная. Транспортные средства подъезжают к перекрестку со всех четырех направлений с периодом, равномерно распределенным в диапазоне a_1...b_1 (сверху), a_2...b_2 (справа), a_3...b_3 (снизу), a_4...b_4 (слева) секунд. Транспортные средства могут поворачивать на перекрестке влево или вправо. Вероятность этих событий составляет a_5 и a_6 независимо от того, с какой стороны подъезжает транспорт. Остальные машины (1-a_5-a_6) проезжают перекресток, не изменяя направления движения. Если транспорт движется по главной дороге (из двух полос), то поворачивающие машины занимают полосу, наиболее близкую для осуществления маневра (если поворот осуществляется влево, то левую, и наоборот).

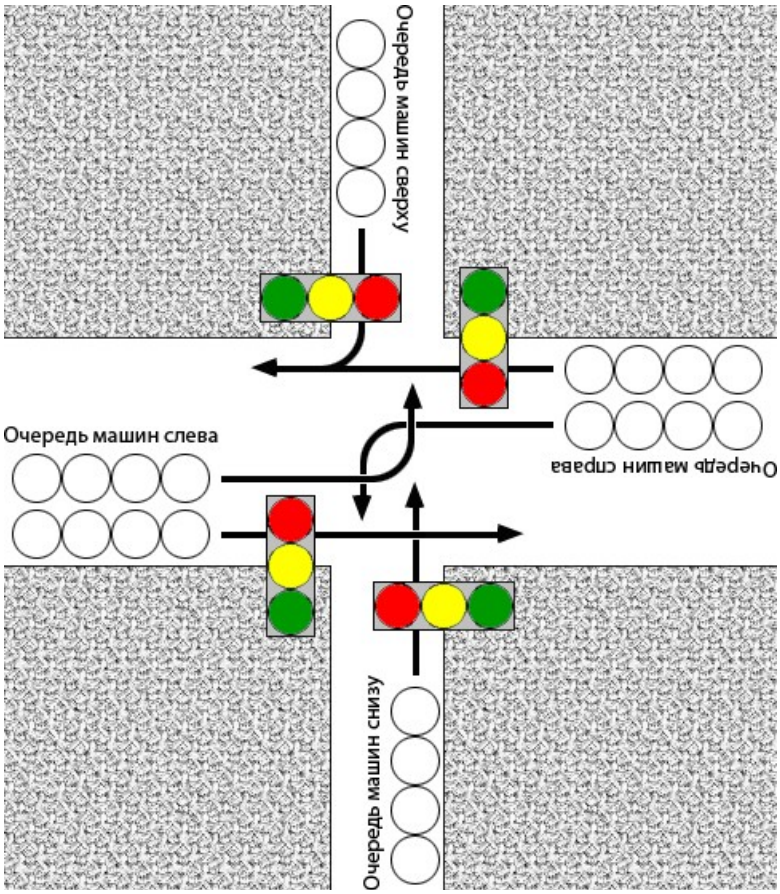
Транспортные средства подразделяются на 4 вида: мопеды, легковые автомобили, грузовые автомобили, автобусы, скорость движения которых на перекрестке постоянна и составляет a_7, a_8, a_9, a_{10} м/с соответственно. Если в очереди к перекрестку или при совершении поворота впереди движется более медленный вид транспорта, то следующие за ним вынуждены снижать скорость, так как обгонять впереди идущего на перекрестке запрещено.

Для регулирования дорожной ситуации на перекрестке установлены 4 светофора на каждое из направлений, включенные попарно. Цикл смены цветов

– стандартный («красный», «желтый», «зеленый», «желтый», далее цикл повторяется). Движение на «желтый» запрещено с двух сторон, но транспорт, начавший маневр (поворот или проезд), может его закончить. В то время, когда горит «зеленый» для главной дороги, для вспомогательной горит «красный», и наоборот. Время переключения светофора на главной дороге составляет a_{11} («красный»), a_{12} («желтый»), a_{13} («зеленый»), a_{12} («желтый») секунд соответственно.

Поворот осуществляется на «красный» (запрещающий) сигнал светофора. Маневры транспортные средства совершают по очереди (например, сначала маневр совершает первое к перекрестку транспортное средство, находящееся на главной дороге слева, затем – на вспомогательной сверху, затем – на главной дороге справа, затем – на вспомогательной снизу. Это может не соответствовать стандартным ПДД, но упрощает программирование и снижает количество коллизий. Однако по усмотрению обучающегося можно учесть приоритеты поворота в соответствии со стандартными правилами ПДД).

Если на дороге образуется пробка (более 50 транспортных средств в одной из очередей соответственно), на дороге появляется регулировщик, а светофоры при этом переводится в аварийное состояние («желтый мигающий»). Роль регулировщика выполняет пользователь, который щелчком мыши по перекрестку может изменить направление движения прямо во время моделирования. Шаг моделирования – 1 секунда.



Визуально, приложение включает область, в которой наглядно отображается процесс моделирования в графическом режиме, область статистики, па-

нель для ввода исходных параметров, кнопка «Старт/останов моделирования» для выполнения шагов моделирования в автоматическом режиме по таймеру, кнопка «Следующий шаг» для ручного выполнения шагов моделирования.

В области статистики должна выводиться следующая информация:

– количество транспортных средств, проехавших через перекрёсток (слева справа, сверху и снизу);

– количество поворачивающих транспортных средств;

– количество транспортных средств в очереди (продумать, как отобразить длинные очереди, которые не помещаются на экране. Допускается, например, вывести только несколько первых машин, а остальные обозначить цифрой, показывающей размер очереди);

– среднее время ожидания в очереди;

– среднее время ожидания на светофоре.

30. Моделирование одного года обучения студентов

Группа из N студентов поступает на обучение на первый курс университета. На первом курсе преподается K_1 экзаменационных, K_2 зачетных дисциплин в осеннем семестре ($K_1 + K_2 <= 10$), K_3 экзаменационных, K_4 зачетных дисциплин в весеннем семестре ($K_3 + K_4 <= 10$). По каждой из дисциплин каждый из студентов обладает начальными знаниями, полученными в школе или самостоятельно, равномерно распределенными в диапазоне $a_1...b_1$%. Каждый студент обладает определенной степенью ответственности, равномерно распределенной в интервале $a_2...b_2$ %, а также умением усваивать новый материал, выраженный значением, равномерно распределенным в интервале $a_3...b_3$ %.

Теоретическая часть каждого семестра длится P_1 недель (обычно 17). В течение этого времени преподавателями университета читается $P_1*(K_1+K_2)$ лекций и проводится столько же лабораторных занятий в осеннем семестре. Соответственно в весеннем семестре читается $P*(K_3+K_4)$ лекций и столько же лабораторных занятий. Экзаменационная сессия после каждого семестра длится P_2 недель, затем следуют каникулы продолжительностью P_3 недель. Как лекционные, так и лабораторные занятия по всем предметам распределены равномерно на одной неделе обучения.

Посещение одного лекционного занятия повышает уровень знаний студента на величину $a_3/P_1...b_3/P_1$ %, по соответствующей дисциплине. Т.е., если слудент посетит все лекционные занятия, то его уровень знаний возрастет на величину $a_3...b_3$. Однако студент может и не прийти на лекцию, если его степень ответственности менее a_4. В этом случае новых знаний он не получает.

На лабораторных занятиях студентам предлагается выполнить M лабораторных работ (одинаковое количество по всем предметам), причем для выполнения каждой из работ необходимо, чтобы студент обладал уровнем знаний не менее величины, равномерно распределенной в диапазоне $a_5...b_5$. Как только студент достигает необходимого уровня знаний, он тут же начинает выполнять соответствующую лабораторную работу (все лабораторные по всем дисциплинам выдаются в начале семестра). Выполнение лабораторной работы занимает по времени $a_6...b_6$ суток.

Как только лабораторная работа готова, студент обязан (на этот раз без пропусков) посетить ближайшее лабораторное занятие по соответствующему предмету, чтобы сдать ее.

По окончании теоретической части семестра подводится промежуточный итог: студенты, выполнившие все лабораторные работы, получают либо зачет для зачетных дисциплин, либо допуск к экзамену для экзаменационных. Остальные студенты отчисляются.

Во время экзаменационной сессии допущенные студенты сдают экзамены, причем для успешной сдачи экзамена необходимо обладать уровнем знаний не менее a_7 %.

На каникулах процесс моделирования приостанавливается (т.е. студенты никуда не перемещаются, занятия не проводятся). Затем моделирование начинается заново для следующего семестра, однако количество дисциплин (K_3 + K_4), начальные знания студентов по этим дисциплинам будут другие. Шаг моделирования – 1 сутки.

Для визуализации процесса обучения предлагается нарисовать в левой части окна $2*(K_1 + K_2)$ или $2*(K_3 + K_4)$ прямоугольников, имитирующих дисциплины (лекции и лабораторные занятия). Названия дисциплин можно хранить в программе в виде массива. Дисциплины, по которым проводятся занятия в текущие сутки (на текущем шаге) подсвечивать или выделять, расписание распределить самостоятельно, главное, чтобы к окончанию времени моделирования теоретической части все $2*(K_1 + K_2)$ или $2*(K_3 + K_4)$ прямоугольников были подсвечены P_1 раз. В прямоугольниках, имитирующих лабораторные занятия, вывести номера лабораторных работ и уровень знаний, необходимый для их выполнения. В прямоугольниках, имитирующих лекции по экзаменационным предметам, вывести процент знаний, необходимый для сдачи экзамена.

Студентов в виде прямоугольников изобразить справа, при этом для каждого вывести все их параметры для всех дисциплин (текущие знания по дисциплинам, степень ответственности. Номера сданных, выполненных, выполняемых и невыполненных лабораторных работ обозначить разными цветами). Посещение студентом лекции или лабораторного занятия обозначить линией или стрелкой.

Визуально, приложение включает область, в которой наглядно отображается процесс моделирования в графическом режиме, область статистики, панель для ввода исходных параметров, кнопка «Старт/останов моделирования» для выполнения шагов моделирования в автоматическом режиме по таймеру, кнопка «Следующий шаг» для ручного выполнения шагов моделирования.

В области статистики должна выводиться следующая информация:

– текущий средний уровень знаний по всем дисциплинам всех студентов;

– количество студентов, пропустивших более 25% теоретического материала;

– количество отчисленных студентов после осеннего и после весеннего семестров;

– среднее количество выполненных лабораторных работ на данный момент;

– количество занятий, на которое пришло менее 50% группы;

– среднее количество выполненных лабораторных работ на студента, но не сданных по причине окончания теоретической части семестра (показатель выводится только после окончания теоретической части семестра, в остальное время моделирования можно выводить прочерк).

7.3. Игровые приложения

31. Игра «Морской бой»

Игра «Морской бой» представляется в виде двух квадратных игровых полей (свое и соперника), разделенных на сектора. Каждый сектор имеет буквенно-цифровые координаты, например «К15» (К-ый столбец, 15-ая строка). В отличие от стандартных правил игры размер игровых полей может быть увеличен вплоть до 30 строк и столбцов и задается в начале игры пользователем.

Каждый игрок имеет одинаковое количество кораблей, которые он расставляет на своем игровом поле перед началом игры. Цель игры стандартная, разгадать поле соперника быстрее, чем это сделает он, «нанося удары» в один из секторов. Для этого игроки по очереди называют координаты сектора, соперник же отвечает фразами «мимо», «ранил», «убил».

Перед началом игры корабли расставляются случайным образом и в случайной ориентации, преследуя определенную игровую стратегию. В стандартных правилах не допускается размешать корабли рядом с друг другом, однако в данной игре это возможно.

В отличие от стандартных правил игры, в игре могут быть задействованы корабли нестандартной формы в форме букв «L», «T», «U», «O», а также в форме креста. Какие корабли и по сколько штук будет задействовано, игроки договариваются перед началом игры. Внешний вид всех допустимых вариантов кораблей показан на рисунке ниже.

N раз за всю игру корабль может телепортировать в свободную (возможно ранее обстрелянную) соперником территорию игрового поля. В этом случае при попадании в корабль соперник отвечает «ранил, телепортирован». Корабль остается раненым, но перемещается в другое место игрового поля, возможно даже вместо ранее убитых кораблей. Однопалубные корабли, занимающие только один сектор, телепортироваться не могут, так как сразу погибают. Телепортироваться может только раненый корабль. Если значение N = 0, то данная модификация игры отключается.

Раненые корабли могут быть отремонтированы в течение K ходов, если соперник их не разгадает и не добьет. Если K = 0, то данная модификация также отключается. Данный режим особенно интересен в сочетании с телепортацией, так как найти вновь раненый корабль сопернику будет затруднительно.

Поскольку игровая стратегия в данной игре несложна, в данном задании предлагается реализовать вариант «человек» – «компьютер». При этом компьютер должен самостоятельно вычислять свой следующий ход на основании предыдущего ответа соперника (случайно компьютер стреляет только если на предыдущих ходах не было раненых кораблей).

Визуальный интерфейс приложения представляет собой два упомянутых игровых поля, на которых отображается основная игровая информация, панель настроек в виде отдельного окна, где можно указать параметры N и K, а также размер игрового поля и количество кораблей разных типов, имя игрока. Панель настроек открывается по нажатию соответствующей кнопки, расположенной внизу игрового поля. Также внизу игрового поля должны присутствовать стандартные кнопки «Начать игру», «Расставить корабли», «Рейтинг игроков» и т.п.

В рейтинге игроков ведется подсчет выигрышей/проигрышей, количества проведенных игр по нестандартным правилам (наличие кораблей нестандартной формы, включены или нет модификации игры). Результат рейтинга должен сохраняться в текстовом файле и отображаться в отдельном окне при нажатии на соответствующую кнопку.

32. Карточная игра «Японский дурак»

Карточная игра «Японский дурак» использует стандартную колоду в 52 карты (без джокеров). Изначально игрокам раздается по N карт (4<=N<=10), остальные карты образуют колоду. Первая снизу карта колоды не пиковой масти (если первая карта снизу колоды имеет пиковую масть, то рассматривается вторая снизу карта и т.п.) представляет козырную масть, и выкладывается лицом вверх перпендикулярно остальным картам колоды. Таким образом, пиковая масть козырем стать не может, так как в этой разновидности игры она имеет особый смысл.

Во многом игра проходит по стандартным правилам карточной игры «Дурак». Начинает игру игрок с минимальным козырем на руках, если такового нет, то рассматриваются простые карты в порядке убывания («туз», «король», «дама»…) который ходит одной или несколькими парными картами. Другой игрок должен покрыть эти карты картами более высокого достоинства или козырем. Отличие заключается в том, что карты пиковой масти могут быть по-

крыты только другими картами пиковой масти более высокого достоинства, но не козырем. Если игрок не может покрыть карты, он их принимает, если может – карты уходят в отбой и больше не участвуют в игре. Далее ходит второй игрок и т.п. После хода игроки берут необходимое число карт из колоды, чтобы довести их количество до N. Цель игры – избавиться от карт на руках раньше соперника (при пустой колоде).

Таким образом, пиковая масть выступает в роли изолированной масти, которую можно покрыть только себе подобным. Пиковый туз не может быть покрыт ничем (как и козырный туз), поэтому ход, включающий в себя одну из этих карт вынуждает соперника взять их.

Карточная игра «Японский дурак» допускает вариации, например, можно подкидывать отбившемуся игроку карты такого же достоинства, но другой масти (включая пику), такая вариация называется «Японский подкидной дурак». Наконец, можно переводить ход на соперника, выложив карту точно такого же достоинства, но другой масти (включая пику). Такая вариация называется «Японский переводной дурак».

В данном задании предлагается реализовать игровое приложение, которое представляет собой компьютерный аналог игры по форме «человек» – «компьютер». Реализация должна включать классический вариант игры и обе описанные модификации. При этом компьютер должен самостоятельно вычислять свой следующий ход из числа допустимых вариантов и реализовывать некоторую выигрышную стратегию.

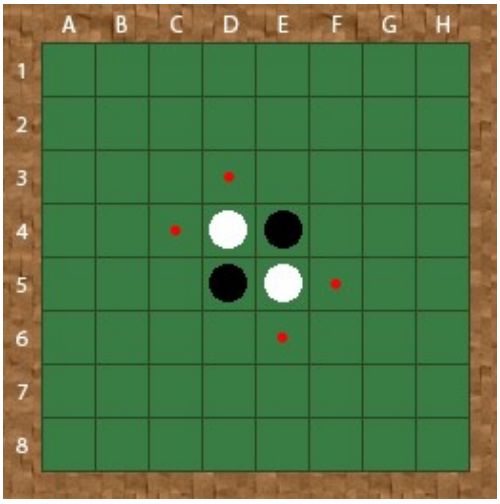
Визуальный интерфейс приложения представляет собой игровое поле, в котором отображаются карты игрока, компьютера (рубашками вверх), колода с козырем. Карты можно изобразить обычными прямоугольниками с цифрой и мастью в углу. Можно и использовать заранее подготовленные изображения. Игровой процесс должен сопровождаться подсказками («бито», «беру» и т.п.). Кроме того, в приложении имеется панель настроек в виде отдельного окна, где можно указать параметр N и выбрать модификацию игры, а также указать имя игрока. Панель настроек открывается по нажатию соответствующей кнопки, расположенной внизу игрового поля. Также внизу игрового поля должны присутствовать стандартные кнопки «Начать игру», «Беру», «Рейтинг игроков» и т.п.

В рейтинге игроков ведется подсчет выигрышей/проигрышей, количества проведенных игр с учетом модификаций игры. Результат рейтинга должен сохраняться в текстовом файле и отображаться в отдельном окне при нажатии на соответствующую кнопку.

33. Игра «Реверси»

Игра «Реверси» (в некоторых источниках называют «Отелло») представляет собой логическую игру на двоих на специальной доске (от шахматной эта доска отличается только тем, что все клетки покрашены в один цвет). Для игры также необходимы специальные фишки, покрашенные с одной стороны в черный, с другой стороны в белый цвет. Количество таких фишек должно соответствовать количеству клеток на доске.

Стандартные правила игры таковы. Ходы делаются по очереди, начинают черные. За один ход игрок должен поставить на доску одну фишку так, чтобы между этой поставленной фишкой и одной из имеющихся уже на доске фишек его цвета находился непрерывный ряд фишек соперника (горизонтальный, вертикальный или диагональный). Иначе говоря, необходимо, чтобы непрерывный ряд фишек соперника оказался «закрыт» фишками текущего игрока с двух сторон. Все фишки соперника, входящие в «закрытый» на этом ходу ряд, меняют цвет (переворачиваются на другую сторону) и переходят к игроку. Иногда перед началом игры в центр доски уже выкладываются по две фишки, как показано на рисунке. Красными точками отмечены возможные ходы черных.



Если в результате одного хода «закрывается» одновременно более одного ряда фишек противника, то переворачиваются все фишки, оказавшиеся на всех «закрытых» рядах. Игрок может выбирать любой из возможных для него ходов. Если игрок имеет возможные ходы, он не может отказаться от хода. Если игрок не имеет допустимых ходов, то ход передается сопернику.

Игра прекращается, доска заполнена фишками полностью или когда ни один из игроков не может сделать ход. По окончании игры проводится подсчет фишек каждого цвета и игрок, чьих фишек на доске выставлено больше, объявляется победителем.

Модификации игры включают:

– игру на нестандартной доске, большего размера, в форме креста или с отверстиями в середине. Если на доске имеются отверстия, то поставить фишку в такие ячейки невозможно. «Закрытый» ряд не может проходить сквозь отверстие;

– помимо обычных фишек, у каждого из игроков имеется N «преданных» фишек. Если выставить такую фишку на доску, то при попадании ее в «закрытый» ряд, она не изменяет цвет, а снимается с доски или ее цвет сохраняется;

– помимо обычных фишек, у каждого из игроков имеется K «взрывающихся» фишек. Если выставить такую фишку на доску, то при попадании ее в «закрытый» ряд, она не изменяет цвет, а взрывается, проделывая отверстие в игровой доске. Такие фишки можно показывать, крестом, поверх цвета, а можно и не показывать вовсе, что приводит к сюрпризам для соперника;

В данном задании предлагается реализовать игровое приложение, которое представляет собой компьютерный аналог игры по форме «человек» – «компьютер». Реализация должна включать классический вариант игры и три описанные модификации. При этом компьютер должен самостоятельно вычислять свой следующий ход из числа допустимых вариантов и реализовывать некоторую выигрышную стратегию. Черными играет (и соответственно первый ходит) либо человек, либо компьютер (выбирается случайно).

Для игр на нестандартной доске (первая модификация) необходимо разработать минимум три игровые доски и сохранить их в текстовом файле (формат хранения придумать самостоятельно). Максимальный размер доски – 50x50 ячеек.

Визуальный интерфейс приложения представляет собой игровую доску с обозначениями, где происходит весь игровой процесс. Кроме того, в приложении имеется панель настроек в виде отдельного окна, где можно указать параметры N, K (если они не равны нулю, то включаются соответствующие модификации. При этом игрок имеет возможность отметить снимать или нет «преданную» фишку с доски – для второй модификации, и показывать ли крест на «взрывающихся» фишках для третьей модификации). Также в панели настроек должна быть возможность загрузить из файла доску. Панель настроек открывается по нажатию соответствующей кнопки, расположенной внизу игрового поля. Также внизу игрового поля должны присутствовать стандартные кнопки «Начать игру», «Рейтинг игроков» и т.п.

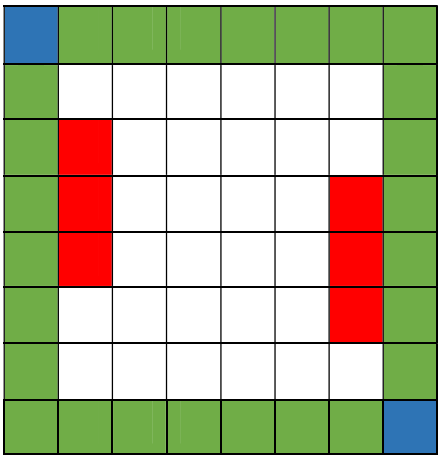
В рейтинге игроков ведется подсчет выигрышей/проигрышей, количества проведенных игр с учетом модификаций игры. Результат рейтинга должен сохраняться в текстовом файле и отображаться в отдельном окне при нажатии на соответствующую кнопку.

34. Игра «Домик»

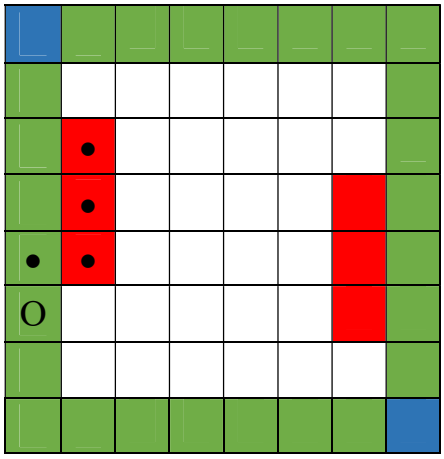
Игра «Домик» представляет собой настольную игру на двоих на специальной доске (см. рисунок). Изначально у каждого из игроков имеется некоторое равное количество фишек, которые находятся в клетках, обозначенных синим цветом. Фишки могут передвигаться по периметру доски только по часовой стрелке (зеленые клетки). Цель игры – провести все фишки в «домик» быстрее соперника. Клетки «домика» обозначены красным цветом. Они должны примыкать к периметру доски, и их должно быть ровно столько же, сколько фишек имеется у игроков изначально. В «домике» фишки располагаются по одной на клетку, в то время как в зеленых и синих клетках доски может располагаться по несколько фишек одного игрока. Белые клетки доски в игре участия не принимают, ставить на них фишки запрещено.

Игра начинается с розыгрыша права первого хода, для чего игроки по очереди кидают игральную кость. Начинает игру тот игрок, который выбросит больше очков (от 1 до 6). Он бросает игральную кость, после чего должен осуществить ход на соответствующее количество клеток любой своей фишкой. Ход пропускается только при отсутствии возможности ходить; в противном случае ходить нужно обязательно и непременно на то количество клеток, кото-

рое выпало на игральной кости. Игрок не может ходить, если соответствующая клетка окажется занятой фишками соперника, однако здесь существуют модификации.



Фишка, которая прошла по всему пути, может зайти в «домик» (перейти на красную клетку), но может и не заходить, а отправиться на второй круг. Фишка, которая зашла в «домик», выбраться обратно (на зеленые клетки) уже не может, но может перемещаться по домику, занимая нужное место и освобождая место для последующих фишек. Пойти на второй круг фишка может либо в соответствии с некоторой игровой стратегией, либо просто потому, что другого хода не существует. Например, если игровая ситуация сложилась так, как показано на рисунке ниже, а игроку выпало «5» или «6» на игральной кости, то завести фишку в домик он не может, так как предполагаемый путь максимум состоит только из четырех клеток (напомним, что ход делается обязательно).



Модификации игры включают:

– игру на нестандартной доске (в форме креста, ломаной линии и т.п.), причем «домик» может включать несколько рядов и строк, к примеру 2x2;

– игра с выбиванием, в которой игрок может выбить фишку соперника, если она одна в зеленой ячейке. Выбитая фишка переносится в начало (синяя ячейка и начинает свой путь заново.

В данном задании предлагается реализовать игровое приложение, которое представляет собой компьютерный аналог игры по форме «человек» – «компь-

ютер». Реализация должна включать классический вариант игры и две описанные модификации. При этом компьютер должен самостоятельно вычислять свой следующий ход из числа допустимых вариантов и реализовывать некоторую выигрышную стратегию. Для игры необходимо разработать минимум три игровые доски и сохранить их в текстовом файле (формат хранения придумать самостоятельно). Максимальный размер доски – 50x50 ячеек. Количество фишек, выдаваемых игрокам нигде не задается, оно вычисляется по числу красных фишек игровой доски.

Визуальный интерфейс приложения представляет собой игровую доску с обозначениями, где происходит весь игровой процесс. Кроме того, в приложении имеется панель настроек в виде отдельного окна, где можно выбрать игровую доску и указать тип игры (с выбиванием или без). Панель настроек открывается по нажатию соответствующей кнопки, расположенной внизу игрового поля. Также внизу игрового поля должны присутствовать стандартные кнопки «Начать игру», «Рейтинг игроков» и т.п.

В рейтинге игроков ведется подсчет выигрышей/проигрышей, количества проведенных игр с учетом модификаций игры. Результат рейтинга должен сохраняться в текстовом файле и отображаться в отдельном окне при нажатии на соответствующую кнопку.

35. Игра «Шахматы»

Игра «Шахматы» представляет собой настольную логическую игру на двоих. Правила игры известны многим, так как игра популярна до сих пор. Игра происходит на доске, разделённой на равные квадратные клетки, или поля. Размер доски – 8×8 клеток. У игроков в начале игры имеется по одинаковому набору фигур. Фигуры одного из игроков условно называются «белыми», другого – «чёрными». В каждый комплект фигур входят: король, ферзь, две ладьи, два слона, два коня и восемь пешек. Белые занимают первую и вторую горизонтали, чёрные – седьмую и восьмую. Пешки расположены на второй и седьмой горизонталях соответственно.

Игроки поочерёдно делают ходы, причем первый ход делают белые. За исключением рокировки, ход заключается в том, что игрок перемещает одну из своих фигур на другое поле по следующим правилам:

Фигуры (кроме коня) передвигаются по прямой линии, при этом все промежуточные поля между начальным и конечным должны быть свободны. Ход на поле, занятое своей фигурой, невозможен. При ходе на поле, занятое чужой фигурой, она снимается с доски (взятие). Король ходит на соседнюю клетку по вертикали, горизонтали или диагонали. Ферзь ходит на любое расстояние по вертикали, горизонтали или диагонали. Ладья ходит на любое расстояние по вертикали или горизонтали. Слон ходит на любое расстояние по диагонали. Конь двигается на две клетки по вертикали и затем на одну клетку по горизонтали, или наоборот, на две клетки по горизонтали и на одну клетку по вертикали, тем самым движение коня напоминает заглавную букву «Г». Пешка может ходить только вперёд (направлением «вперёд» называется направление к восьмой горизонтали для белых или к первой для чёрных): без взятия – на одно по-

ле вперёд по вертикали, а со взятием – по диагонали. Если пешка находится на начальном поле (вторая горизонталь для белых и седьмая для чёрных), то она также может сделать ход без взятия на два поля вперёд. Если пешка доходит до последней горизонтали, она заменяется по выбору игрока на любую другую фигуру того же цвета, кроме короля.

Игра завершается выигрышем одного из соперников или ничьей. Выигрыш (мат королю соперника) возникает, когда король находится под боем, а все клетки вокруг, куда бы он мог отступить, также контролируются. Так же при мате нет возможности сбить угрожающую королю фигуру. Ничья (пат) возникает тогда, когда игрок, имеющий право хода, не может им воспользоваться, так как все его фигуры и пешки (кроме короля) либо сбиты, либо лишены возможности сделать ход по правилам, причём король не находится под шахом.

Разработка компьютерного алгоритма, обеспечивающего выигрышную стратегию для данной игры, является достаточно сложным делом. Игра отличается большим разнообразием ходов, которое лишь возрастает при выполнении расчетов на несколько ходов вперед. Подобные алгоритмы разрабатываются с применением эвристического анализа и самообучения, при котором запоминаются выигрышные стратегии, вычисленные в предыдущих партиях. В связи с этим, в данном задании требуется разработать компьютерный вариант игры по форме «человек» – «человек». В разрабатываемом приложении необходимо предусмотреть следующие возможности:

– поочередная игра за одним компьютером для двух человек (с поворотом доски);

– выбор фигуры вместо пешки, дошедшей до конца;

– проверка корректности хода (программа не позволяет сделать некорректный ход, например пойти произвольной фигурой, когда объявлен шах королю, также необходимо учесть рокировку);

– получение списка всех возможных ходов (в отдельном окне выводится список всех допустимых ходов в стандартной шахматной нотации для сложившейся на доске ситуации);

– проверка шаха королю (выводится сообщение, если это произошло);

– поиск мата в один ход (выводится сообщение, если это возможно);

– поиск мата в два хода (выводится сообщение, если это возможно);

– загрузка и сохранение партии в текстовом файле в стандартной шахматной нотации;

– отсчет времени для каждого из игроков.

Визуальный интерфейс приложения представляет собой игровую доску с обозначениями, где происходит весь игровой процесс. Кроме того, в приложении имеется панель настроек в которой задаются имена игроков, признак игры на время. Панель настроек открывается по нажатию соответствующей кнопки, расположенной внизу игрового поля. Также внизу игрового поля должны присутствовать стандартные кнопки «Начать игру», «Рейтинг игроков» и т.п.

В рейтинге игроков ведется подсчет выигрышей/проигрышей, количества проведенных игр. Результат рейтинга должен сохраняться в текстовом файле и отображаться в отдельном окне при нажатии на соответствующую кнопку.

36. Игра «Пятнашки»

Классическая «игра в 15» или «пятнашки» представляет собой логическую головоломку для одного человека. Традиционно игра производится на специальной доске, разделенной на 16 квадратных клеток. У игрока имеется 15 фишек, размеры которых в точности соответствуют размерам клеток и которые пронумерованы цифрами от 1 до 15. В начале игры фишки выкладываются на доску случайным образом, по одной фишке на клетку. Одна клетка остается пустой.

Ход заключается в том, чтобы переместить одну из соседних с пустой клеткой фишек на эту клетку. Цель игры заключается в том, чтобы путем перемещений упорядочить все фишки по возрастанию их номеров, т.е. от 1 до 15.

Анализ игры (подробно описанный в специализированной литературе) показывает, что любая случайная комбинация фишек в конечном итоге может быть упорядочена, но за исключением двух последних фишек. Т.е. исход игры может привести либо к комбинации «13», «14», «15» в последнем ряду, либо к комбинации «13», «15», «14». Первая комбинация является выигрышной. Доказано, что получить из второй комбинации первую невозможно. Поэтому вероятность выиграть в «пятнашки» составляет 50%.

Модификации игры включают:

– увеличение размеров игрового поля (и соответствующего числа фишек), использование доски нестандартной формы, в т.ч. с отверстиями (такие игры не анализировались, и играть в них довольно интересно);

– изменение условия победы, при котором цифры собираются не по строкам («слева-направо, сверху-вниз»), как в классическом варианте, а по столбцам (сверху-вниз», «слева-направо);

– составление некоторого слова или заранее заданной фразы, для чего на фишках вместо цифр изображаются буквы, из которых состоит эта фраза. В этой модификации должно также существовать несколько пустых фишек, имитирующих пробелы;

– составление некоторого, заранее заданного в виде миниатюры изображения. В этом случае фишки заливаются цветами (или частями изображения), так чтобы после упорядочивания на игровом поле образовалось изображение.

В данном задании предлагается реализовать игровое приложение, которое представляет собой компьютерный аналог игры. Реализация должна включать классический вариант игры и четыре описанные модификации. Для игры необходимо разработать минимум три игровые доски и сохранить их в текстовом файле (формат хранения придумать самостоятельно). Максимальный размер доски – 50x50 ячеек. Также необходимо предусмотреть подсчет числа выполненных ходов и времени, потраченного на игру.

Визуальный интерфейс приложения представляет собой игровую доску, где происходит весь игровой процесс. Кроме того, в приложении имеется панель настроек в виде отдельного окна, где можно выбирать модификацию игры, задавать имя игрока, выбирать игровую доску, вводить слово или фразу для третьей модификации. Панель настроек открывается по нажатию соответствующей кнопки, расположенной внизу игрового поля. Также внизу игрового

поля должны присутствовать стандартные кнопки «Начать игру», «Рейтинг игроков» и т.п.

В рейтинге игроков ведется подсчет количества проведенных игр, времени, затраченного на игру, и числа ходов. Результат рейтинга должен сохраняться в текстовом файле и отображаться в отдельном окне при нажатии на соответствующую кнопку.

37. Игра «Змейка»

Игра «Змейка» представляет собой классическую казуальную компьютерную игру, рассчитанную на одного человека. Традиционно игровое поле представляет собой прямоугольное или квадратное поле, разбитый на клетки, в котором с определенной скоростью перемещается «змейка». Она представляет собой объект, занимающий несколько клеток, перемещение «змейки» в определенном направлении обычно представляет собой добавление новой клетки со стороны «головы» и удаление клетки со стороны «хвоста». Таким образом, длина «змейки» не изменяется, но сама она смещается на одну клетку по направлению своего движения. Происходит это автоматически по таймеру, задающему скорость игры.

Изначально «змейка» состоит из трех клеток. Игрок управляет «змейкой», изменяя направление ее движения. «Змейка» может как угодно изгибаться, но она не должна врезаться сама в себя, либо в границы игрового поля. Кроме того на поле имеется «фрукт», поедая который, «змейка» вырастает на одну единицу длины. Для поедания «фрукта» «змейка» должна наехать на него «головой». После поедания новый «фрукт» появляется на игровом поле в случайном месте, но не внутри «змейки». При поедании «фрукта» игроку добавляются очки. Цель игры заключается в том, чтобы набрать как можно больше очков.

Если «змейка» врезается сама в себя или в границы поля, то в зависимости от модификации игры она либо погибает сразу, либо ее длина начинает уменьшаться, давая возможность игроку «отвернуть в сторону».

Модификации классического варианта игры включают:

– увеличение размеров игрового поля, нестандартная форма поля (например в виде креста или зигзага, что придает дополнительный интерес игре;

– бесконечное поле, в котором «змейка» не врезается в границы поля, а появляется с его противоположной стороны;

– наличие препятствий на игровом поле, в которые «змейка» не может врезаться (при этом ее длина либо уменьшается, либо она сразу погибает);

– наличие пробиваемых препятствий на игровом поле. Если «змейка» врезается в такое препятствие, то ее длина уменьшается, но и препятствие исчезает;

– увеличение скорости игры после набора игроком определенного количества очков;

– наличие на поле телепортов, въезжая в который, «змейка» (точнее ее часть) начинает выползать из другой (ответной) части телепорта;

– наличие на поле нескольких «фруктов» одновременно и «фруктов», увеличивающих длину «змейки» на большее количество ячеек.

В данном задании предлагается реализовать приложение, которое представляет собой компьютерную игру по описанным правилам. Реализация должна включать классический вариант игры и все описанные модификации. Для игры необходимо разработать минимум пять игровых полей и сохранить их в текстовом файле (формат хранения придумать самостоятельно). Максимальный размер поля – 100x100 ячеек. Также необходимо предусмотреть подсчет числа очков и времени, потраченного на игру.

Визуальный интерфейс приложения представляет собой игровое поле, где происходит весь игровой процесс. Кроме того, в приложении имеется панель настроек в виде отдельного окна, где можно выбирать модификацию игры, задавать имя игрока, выбирать игровое поле, флажок бесконечного поля, поля для ввода количества препятствий и пробиваемых препятствий (которые потом генерируются случайно), флажок для увеличения скорости игры, флажок наличия на карте телепортов (которые также расставляются случайно). Панель настроек открывается по нажатию соответствующей кнопки, расположенной внизу игрового поля. Также внизу игрового поля должны присутствовать стандартные кнопки «Начать игру», «Рейтинг игроков» и т.п.

В рейтинге игроков ведется подсчет количества проведенных игр, набранных очков и времени, затраченного на игру. Результат рейтинга должен сохраняться в текстовом файле и отображаться в отдельном окне при нажатии на соответствующую кнопку.

38. Игра «Арканоид»

Игра «Арканоид» представляет собой казуальную компьютерную игру, рассчитанную на одного игрока. Игровое поле представляет собой прямоугольник, разбитый на квадратные ячейки. В верхней части поля находится несколько элементов, которые необходимо разбить при помощи шарика. Для разбития элемента шарик должен его коснуться. За разбитие элементов игроку начисляются очки. Разбитые элементы удаляется с игрового поля. Цель игрового уровня (или всей игры) – разбить все элементы.

Движение шарика начинается с подвижной платформы, расположенной внизу игрового поля, перемещением которой управляет игрок. Шарик начинает движение в направлении, угол которого зависит от его местоположения на платформе. Так, чем ближе шарик к краю платформы, тем острее угол. Это же правило действует и при последующих отскоках шарика от платформы (см. далее). Во время движения шарик может многократно отскакивать от боковых и верхней стенок игрового поля или разбиваемых элементов по принципу «угол падения равен углу отражения». Однако шарик не может отскакивать от нижней стенки, там где находится подвижная платформа. Когда шарик направляется вниз, игрок должен успеть подвести платформу и отбить ею шарик. Если игрок не успевает это сделать, игра завершается.

Модификации классического варианта игры включают:

– наличие на игровом поле элементов, которые разбиваются не с первого раза, а с нескольких попыток. Таких элементов может быть несколько типов, например, одни пробиваются с пятой попытки, другие с десятой и т.п. Остав-

шееся число попыток, после которого элемент разобьется, можно отображать внутри элемента цифрой, а сами элементы раскрашивать в разные цвета.

– наличие на игровом поле неразбиваемых элементов. Такие элементы не требуется разбивать, чтобы закончить уровень, но они также влияют на траекторию движения шарика и способны в некоторых случаях усложнить или, наоборот, упростить прохождение;

– «огненный шарик», который пролетает сквозь все элементы не меняя своей траектории, и отражаясь лишь от стенок. Такой шарик прожигает все элементы на своем пути, включая неразбиваемые. Способность дается на короткое время в виде бонуса;

– наличие бонусов, которые выпадают из разбиваемых элементов с определенной долей вероятности. Игрок должен поймать их платформой. Среди бонусов могут встречаться: 1) увеличение размеров платформы, 2) досрочный выход из уровня, 3) дополнительная жизнь, 4) «огненный шарик», 5) начисление дополнительных очков, 6) «прилипающий шарик» (прилипает к платформе при возвращении вниз, а не отбивается сразу. Это дает время пользователю подвинуть платформу, чтобы направить шарик по нужной траектории).

В данном задании предлагается реализовать приложение, которое представляет собой компьютерную игру по описанным правилам. Реализация должна включать классический вариант игры и все описанные модификации. Для игры необходимо разработать минимум 10 уровней для демонстрации работы приложения. Уровни могут храниться в текстовых файлах (формат хранения придумать самостоятельно).

Визуальный интерфейс приложения представляет собой игровое поле, где происходит весь игровой процесс. Кроме того, в приложении имеется панель настроек в виде отдельного окна, где можно задавать имя игрока, выбирать сложность игры (влияет только на скорость движения шарика). Панель настроек открывается по нажатию соответствующей кнопки, расположенной внизу игрового поля. Также внизу игрового поля должны присутствовать стандартные кнопки «Начать игру», «Рейтинг игроков» и т.п.

В рейтинге игроков ведется подсчет количества проведенных игр, набранных очков и времени, затраченного на игру. Результат рейтинга должен сохраняться в текстовом файле и отображаться в отдельном окне при нажатии на соответствующую кнопку.

39. Игра «Поймай луч»

Игра «Поймай луч» представляет собой логическую компьютерную игру, рассчитанную на одного игрока. Игровое поле представляет собой прямоугольник, разбитый на квадратные ячейки. На игровом поле находятся:

– один или несколько лазеров;

– приёмники лазерного излучения;

– одно или несколько зеркал.

Лазеры излучают когерентный пучок света, который на игровом поле отображается в виде тонкой линии. Приемники соответственно улавливают этот луч. Цель игры – направить лучи от всех лазеров на приемники при помо-

щи зеркал, которые могут поворачиваться на угол, кратный 90. Все игровые элементы закреплены неподвижно, т.е. их нельзя перемещать, а лазеры и приемники нельзя еще и поворачивать. Поворот зеркала осуществляется при помощи щелчка мышки по нему. Зеркало может отражать лазерный луч только одной стороной.

Модификации классического варианта игры включают:

– наличие на игровом поле лазеров и приемников разных цветов, при этом разумеется каждый приемник может поймать только луч своего цвета;

– наличие на игровом поле непроходимых для лазерного луча препятствий, которые просто обрывают луч;

– наличие на игровом поле полупрозрачных зеркал, призм и зеркальных кубов, разделяющих лазерный луч на несколько равных по мощности лучей или, наоборот фокусирующих из нескольких пучков света один;

– лазеры могут иметь параметр мощности. Например один лазер может светить с мощностью 50% красным цветом, другой – мощностью 100% зеленым цветом. Приемник же для своей активации может требовать излучения разных цветов и разной мощности, в том числе смешанных, например 25% зеленого и 25% красного. При этом если осветить такой приемник лучом большей мощности, лишняя энергия рассеивается без вреда для него. В этом случае все полупрозрачные зеркала, призмы и зеркальные кубы из предыдущей модификации, разделяя или собирая луч, понижают или повышают его мощность пропорционально количеству образуемых пучков света.

В данном задании предлагается реализовать приложение, которое представляет собой компьютерную игру по описанным правилам. Реализация должна включать классический вариант игры и все описанные модификации. Для игры необходимо разработать минимум 10 уровней для демонстрации работы приложения. Уровни могут храниться в текстовых файлах (формат хранения придумать самостоятельно).

Визуальный интерфейс приложения представляет собой игровое поле, где происходит весь игровой процесс. Кроме того, в приложении имеется панель настроек в виде отдельного окна, где можно задавать имя игрока, выбирать сложность игры (влияет только на применяемые модификации). Панель настроек открывается по нажатию соответствующей кнопки, расположенной внизу игрового поля. Также внизу игрового поля должны присутствовать стандартные кнопки «Начать игру», «Рейтинг игроков» и т.п.

В рейтинге игроков ведется подсчет количества проведенных игр, набранных очков и времени, затраченного на игру. Результат рейтинга должен сохраняться в текстовом файле и отображаться в отдельном окне при нажатии на соответствующую кнопку.

40. Игра «Сокобан»

Игра «Сокобан» (в русском переводе – кладовщик) представляет собой логическую компьютерную игру, рассчитанную на одного игрока. Игровое поле представляет собой прямоугольник, разбитый на квадратные ячейки. На игровом поле находятся ящики, места размещения для этих ящиков, а также иг-

ровой персонаж – сам кладовщик. Игрок управляет этим персонажем, перемещая его по полю во все четыре стороны, шаг перемещения соответствует одной ячейке игрового поля. Кладовщик может перемещать ящики, находящиеся на поле, толкая их впереди себя. В классическом варианте он не способен толкать несколько ящиков сразу. Цель игры – разместить все ящики на соответствующих местах игрового поля за минимальное число ходов. Ход заключается в перемещении кладовщика из клетки в клетку (в т.ч. и тогда, когда он не толкает ящик). За успешное перемещение ящика игроку начисляются очки.

Помимо ящиков и мест для их размещения в классическом варианте также встречаются и стены – неперемещаемые элементы. Если какой либо ящик находится рядом со стеной, то перемещать его можно только вдоль этой стены, так как у кладовщика нет возможности попасть между стеной и ящиком, а тянуть ящик на себя он не может. Если же ящик загнать в угол, то он окажется запертым вовсе, после чего, вероятно, игру придется начать сначала.

Модификации классического варианта игры включают:

– нестандартная форма поля (например в виде креста или зигзага, что придает дополнительный интерес игре;

– наличие на игровом поле дверей, открывающихся только в одну сторону. Это позволяет создавать односторонние проходы и дополнительно усложняет игру;

– на ящиках и местах размещения ящиков, расположенных на игровом поле, могут быть номера. В этом случае от игрока требуется переместить каждый ящик на свое место в соответствии с номером;

– на поле могут находиться бутыли с «эликсиром силы». Выпив такой бутыль (просто наехав на него), кладовщик приобретает способность толкать не по одному, а по несколько ящиков сразу, но не более нескольких ходов;

– наличие на игровом поле телепортов, позволяющих мгновенно перемещать как самого кладовщика, так и толкаемые им ящики в другое место.

В данном задании предлагается реализовать приложение, которое представляет собой компьютерную игру по описанным правилам. Реализация должна включать классический вариант игры и все описанные модификации. Для игры необходимо разработать минимум 10 уровней для демонстрации работы приложения. Уровни могут храниться в текстовых файлах (формат хранения придумать самостоятельно).

Визуальный интерфейс приложения представляет собой игровое поле, где происходит весь игровой процесс. Кроме того, в приложении имеется панель настроек в виде отдельного окна, где можно задавать имя игрока, выбирать сложность игры (влияет только на применяемые модификации). Панель настроек открывается по нажатию соответствующей кнопки, расположенной внизу игрового поля. Также внизу игрового поля должны присутствовать стандартные кнопки «Начать игру», «Рейтинг игроков» и т.п.

В рейтинге игроков ведется подсчет количества проведенных игр, набранных очков и сделанных ходов. Результат рейтинга должен сохраняться в текстовом файле и отображаться в отдельном окне при нажатии на кнопку.

41. Игра «Электрик»

Игра «Электрик» представляет собой казуальную логическую компьютерную игру, рассчитанную на одного игрока. Игровое поле представляет собой прямоугольник, разбитый на квадратные ячейки. На игровом поле находятся электрические розетки (источники тока) и лампочки (потребители тока). Перемещаться по полю указанные элементы не могут.

Цель игры заключается в том, чтобы соединить каждую розетку с лампочкой при помощи проводов, имеющихся в инвентаре у игрока. Провод (с вилками на обоих концах) имеет определенную длину и может быть уложен только вдоль ячеек игрового поля, поэтому его длина измеряется в ячейках. Если длина провода меньше, чем расстояние между соответствующими источником и потребителем, то таким проводом соединить лампочку с розеткой невозможно. Если провод длиннее, чем требуется, то электрик может скрутить остаток провода в моток и положить его рядом с розеткой. При укладывании провода могут пересекаться. За каждое удачное подключение пользователю начисляются очки.

Модификации классического варианта игры включают:

– нестандартная форма игрового поля (например в виде креста или зигзага, что придает дополнительный интерес игре;

– наличие на игровом поле стен и других препятствий, не позволяющих проложить провод напрямую, а только в обход;

– наличие в инвентаре у игрока, помимо проводов удлинителей и разветвителей электрического тока. Указанные приспособления состоят из провода фиксированной длины с вилкой и одной розетки (удлинители) или нескольких розеток (разветвители);

– наличие в инвентаре у игрока дрели и нескольких попыток просверлить стену (прежде чем сверло сломается). Если просверлить стену, то провод можно пропустить через отверстие. Данная модификация актуальна, если стены вообще есть на поле;

– наличие в инвентаре у игрока плоскогубцев и изоленты. Если они есть, то игрок может скрутить два провода вместе, получив один провод суммарной длины. Этот вариант отличается от удлинителей тем, что скрученные провода разделить обратно уже невозможно;

– розетки и лампочки могут быть разной мощности, а потому подключить к одной розетке несколько лампочек, имеющих суммарную мощность более чем допускается для этой розетки невозможно.

Длину провода рекомендуется выводить у каждого провода, имеющегося в инвентаре электрика, а для розеток и лампочек выводить максимально допустимую и потребляемую мощности для наглядности.

В данном задании предлагается реализовать приложение, которое представляет собой компьютерную игру по описанным правилам. Реализация должна включать классический вариант игры и все описанные модификации. Для игры необходимо разработать минимум 10 уровней для демонстрации работы приложения. Уровни могут храниться в текстовых файлах (формат хранения придумать самостоятельно).

Визуальный интерфейс приложения представляет собой игровое поле, где происходит весь игровой процесс, инвентаря электрика, где находятся игровые элементы. Кроме того, в приложении имеется панель настроек в виде отдельного окна, где можно задавать имя игрока, выбирать сложность игры (влияет только на применяемые модификации). Панель настроек открывается по нажатию соответствующей кнопки, расположенной внизу игрового поля. Также внизу игрового поля должны присутствовать стандартные кнопки «Начать игру», «Рейтинг игроков» и т.п.

В рейтинге игроков ведется подсчет количества проведенных игр и набранных очков. Результат рейтинга должен сохраняться в текстовом файле и отображаться в отдельном окне при нажатии на кнопку.

42. Игра «Поле чудес»

Игра «Поле чудес» представляет собой аналог известной телевизионной игры. В начале раунда ведущим озвучивается задание, описывающее скрытое слово или фразу. Цель игры заключается в том, чтобы угадать раньше соперников это слово. Перед игроками имеется табло с зашифрованным словом и барабан. Барабан состоит из множества секторов, на которых имеются цифры – число зарабатываемых очков при выпадении сектора. Над барабаном неподвижно закреплена стрелка. Игроки вращают барабан по очереди. Когда барабан остановится, стрелка укажет на один из секторов – это и будет количество очков. Очки начисляются игроку только в том случае, если он угадает одну из букв зашифрованного слова. Если в слове несколько одинаковых букв, то выпавшие на барабане очки умножаются на количество угаданных букв. Например, если игрок назовет букву «Е» для зашифрованной на табло фразы «ПОЛЕ ЧУДЕС», то число выпавших очков умножается на 2, так как буква «Е» встречается в зашифрованной фразе дважды. Угаданные буквы зашифрованного слова отображаются на табло.

В любой момент игрок может попытаться угадать слово, если попытка окажется успешной, раунд завершается его победой, если нет, игрок выбывает из игры. Игровой раунд также завершается победой, если угаданы все буквы зашифрованного слова.

В раунде участвует три игрока. Один из них человек, еще двое – компьютерные соперники. Место за барабаном игрока-человека определяется при помощи жеребьевки, проводимой перед началом раунда.

Компьютерные соперники могут иметь разный интеллект и разные алгоритмы выбора буквы, которую следует называть, например:

– назвать случайную букву русского алфавита;

– назвать букву, по статистике наиболее часто встречаемую среди слов русского языка, например, сначала популярные гласные, затем популярные согласные, после чего прочие буквы, например в порядке («а», «о», «е», «и», «п», «р» и т.д.);

– назвать букву, опираясь на внутренний словарный запас. Это самый сложный вариант реализации, для которого компьютерному игроку нужен собственный словарь (например в виде текстового файла). Из этого словаря пер-

вым делом выбираются все слова, длина которых соответствует зашифрованному слову. Затем среди этих слов подсчитывается частота вхождения букв и определяется самая популярная буква. Если слово уже содержит открытые буквы, то слова из словаря на первом этапе необходимо выбирать с учетом этого. Если зашифрованное слово не совпадает со словом в словаре игрока (такое вполне может быть, компьютерный соперник не обязан «знать» все зашифрованные слова), то игрок действует либо по первому, либо по второму варианту.

Для последних двух указанных вариантов реализации компьютерных соперников необходимо предусмотреть модификацию, при которой с определенной вероятностью соперник иногда ошибается, и называет «не ту» букву. Это необходимо сделать для того, чтобы у реального игрока (человека) был шанс на победу.

Помимо секторов с изображением очков, на барабане имеются и другие, специальные, сектора:

– «Б» – банкрот, все очки игрока обнуляются, ход передается следующему игроку;

– «ПРИЗ» – игрок может получить специальный приз и покинуть игру (реализовывать торг с ведущим, как в телевизионной версии игры, необязательно), но может и отказаться от приза, продолжив игру. В последнем случае он получает 1000 очков, умноженных на количество угаданных одинаковых букв зашифрованного слова;

– «+» – игрок по своему желанию может открыть любую букву зашифрованного слова. Если таких букв окажется несколько одинаковых, то они тоже открываются, а игрок получает 1000 очков, умноженных на их количество;

– «х2» – все очки игрока умножаются на два. При этом, если он угадает несколько букв, то очки умножаются на 2.5, 3, 3.5 и т.д. в зависимости от числа угаданных букв.

Помимо этого, в игре имеется режим «Две шкатулки» – дополнительный приз в 5000 очков за три подряд угаданные одним игроком буквы. Игрок (в т.ч. компьютерный) должен выбрать одну из них и в зависимости от этого выбора ему либо начисляются дополнительные очки, либо нет.

Перечень угадываемых слов (и заданий к ним) должен храниться в текстовом файле. Задание и зашифрованное слово выбирается случайно из этого перечня.

Визуальный интерфейс приложения представляет собой игровое поле, где происходит весь игровой процесс (необходимо изобразить минимум барабан и табло). Кроме того, в приложении имеется панель настроек в виде отдельного окна, где можно задавать имя игрока, выбирать сложность игры (влияет только на компьютерных соперников, например при слабом уровне игры соперники выбираются из числа тех, что называют буквы случайно, а при сложном – те, что используют встроенный словарь). Панель настроек открывается по нажатию соответствующей кнопки, расположенной внизу игрового поля. Также внизу игрового поля должны присутствовать стандартные кнопки «Начать раунд», «Взять приз», «Сказать слово», «Рейтинг игроков» и т.п.

В рейтинге игроков ведется подсчет количества проведенных игр и набранных игроками очков в зависимости от уровня сложности. Результат рейтинга должен сохраняться в текстовом файле и отображаться в отдельном окне при нажатии на кнопку.

43. Игра «Тетрис»

Игра «Тетрис» представляет собой казуальную компьютерную игру, рассчитанную на одного игрока. Игровое поле представляет собой прямоугольник, разбитый на квадратные ячейки, своего рода «стакан», в который сверху непрерывно опускаются геометрические фигуры – тетрамино. Этот термин означает, что фигура может состоять не более чем из четырех блоков (тетра – четыре). Размеры каждого блока соответствуют размеру ячейки игрового поля. Фигура опускается с постоянной скоростью до тех пор, пока не упрется в дно «стакана» или фигуры, ранее упавшие в этот «стакан». После этого сверху «стакана» появляется новая, случайная, фигура. Размеры «стакана» составляют 10х20 ячеек.

Пока фигура опускается на дно «стакана», игрок может двигать ее влево-вправо, а также поворачивать на угол, кратный 90°. Цель этих манипуляций – уложить фигуру в «стакан» таким образом, чтобы образовался один или несколько заполненных горизонтальных рядов. Полностью заполненные горизонтальные ряды удаляются из «стакана», а все элементы, находящиеся выше, автоматически сдвигаются вниз, освобождая место. Игроку при этом начисляются очки, причем чем больше заполненных горизонтальных рядов исчезает после установки текущей фигуры, тем больше начисляется очков. Так, если исчезает только один ряд, игроку начисляется 100 очков, если два 300 очков, если три – 700, и если четыре –1500 очков. Больше рядов исчезнуть не может, так как фигура максимум может состоять только из четырех блоков. Игрок также может ускорить движение фигуры по стакану, если он уже определился с ее местоположением.

По сути, в классической версии игры нет условия победы – процесс укладывания фигур может происходить достаточно долго. Проигрывает же игрок тогда, когда очередную фигуру положить уже некуда – «стакан» заполнен полностью не до конца заполненными рядами.

В некоторых случаях игровое поле содержит окно, в котором отображается миниатюра фигуры, которая выпадет следующей.

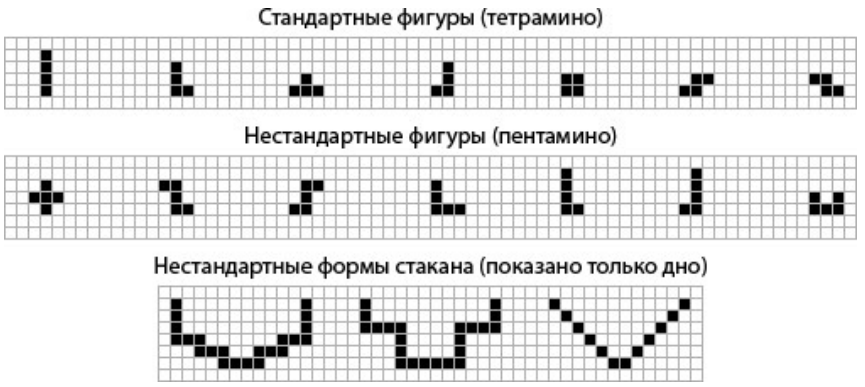
Модификации классического варианта игры включают:

– наличие нестандартных геометрических фигур из пяти блоков – пентамино, показанных на рисунке;

– нестандартная форма «стакана». Дно «стакана» не плоское, как в классической версии игры, а такое, как показано на рисунке. При этом правила игры не меняются – если строка заполнена полностью, она исчезает, а все блоки, находящиеся выше, опускаются;

– некоторые фигуры (размером менее трех блоков) могут проходить сквозь блоки и опускаться в стакан, заполняя собой разрывы в рядах;

– с некоторой долей вероятности выпадает фигура, размеры которой в два раза больше;



В данном задании предлагается реализовать приложение, которое представляет собой компьютерную игру по описанным правилам. Реализация должна включать классический вариант игры и все описанные модификации.

Визуальный интерфейс приложения представляет собой игровое поле, состоящее из «стакана», поля «следующая фигура», поля с числом набранных очков. Кроме того, в приложении имеется панель настроек в виде отдельного окна, где можно задавать имя игрока, выбирать режим игры – форму стакана и допустимые группы выпадающих фигур. Панель настроек открывается по нажатию соответствующей кнопки, расположенной внизу игрового поля. Также внизу игрового поля должны присутствовать стандартные кнопки «Начать игру», «Рейтинг игроков» и т.п.

В рейтинге игроков ведется подсчет количества проведенных игр и набранных очков. Результат рейтинга должен сохраняться в текстовом файле и отображаться в отдельном окне при нажатии на кнопку.

44. Игра «Теннис»

Игра «Теннис» представляет собой казуальную компьютерную игру, рассчитанную на одного или двух игроков (режим «игры с компьютером» или «друг против друга»). В последнем случае игроки играют друг против друга за одним компьютером. Игровое поле представляет собой прямоугольную область, расположенную горизонтально и разделенную на две половины. Слева и справа игрового поля находятся подвижные платформы (своеобразные ракетки), которыми управляют игроки (при помощи кнопок клавиатуры) или компьютер в зависимости от выбранного режима игры. Платформы способны двигаться во всех четырех направлениях (вверх, влево, вправо, вниз), но только на своей половине игрового поля. На поле также имеется теннисный мяч. Игроку необходимо отбивать мяч, подводя платформу в нужное место.

Игра состоит из нескольких раундов. Выигрыш в каждом раунде приносит игроку одно очко. Для победы необходимо выиграть определенное количество раундов (задается заранее перед началом игры). Для победы в раунде необходимо отбивать мячи, причем так, чтобы мяч не вылетел за пределы игрового поля. Если мяч вылетает за боковые границы игрового поля на своей половине или половине соперника, это означает, что соперник не может его отбить, и победа в раунде достается сопернику. Если мяч способен пролететь половину поля соперника насквозь и тот его не отобьет, то победа в раунде достается те-

кущему игроку. Мячи могут отбиваться только платформами (по принципу «угол падения равен углу отражения»), от стенок игрового поля они не отскакивают.

Движение теннисного мяча начинается с одной из платформ (подача). В начале игры подающий игрок определяется методом жеребьевки. Последующие подачи выполняет игрок, проигравший предыдущий раунд.

Модификации классического варианта игры включают:

– наличие препятствий, расположенных ближе к центру игрового поля. При этом необходимо следить, чтобы препятствия были симметрично расставлены на левой и правой половине поля, чтобы у игроков были равные шансы на победу. От препятствий теннисный мяч может отражаться также, как и от подвижных платформ. Подвижная платформа не может зайти на препятствие;

– наличие бонусов и антибонусов, периодически выпадающих на левой или правой половине игрового поля. Для получения бонуса или антибонуса игрок должен навести на него платформу. Антибонусы, как это понятно из названия, вредят текущему игроку, их разумеется нужно избегать, но можно получить их случайно, наведя на него платформу впопыхах. Среди бонусов могут быть: победа в текущем раунде, увеличение размера платформы на некоторое время, увеличение скорости мяча при отбиве (что усложняет ситуацию для соперника). Среди антибонусов могут встречаться противоположные по смыслу виды: проигрыш в текущем раунде, уменьшение размера платформы на некоторое время, уменьшение скорости мяча при отбиве;

– динамический режим игры, в котором при подаче или отбиве мяч получает некоторое ускорение, а во время движения замедляется (вплоть до полной остановки). При этом, если мяч остановится на половине поля соперника, то он проиграет, даже если траектория мяча направлена так, что он неминуемо вылетел бы за боковые границы поля, будь у него достаточно скорости. Кроме того, мяч может не вылететь на половину поля соперника (если угодно «запутаться в сетке»), если его скорость становится ниже некоторого установленного значения.

Для режима игры с компьютером необходимо также реализовать компьютерных соперников, управляющих платформой по определенному алгоритму. Первый вариант компьютерного игрока просто подводит платформу по одной оси координат так, чтобы мяч мог от нее отразиться, не просчитывая при этом последствия отскока. В этом случае мяч легко может оказаться за пределами игрового поля. Второй вариант компьютерного игрока вычисляет траекторию, по которой полетит отскочивший мяч, и чтобы он не вылетал за пределы поля, двигает платформу по обеим осям координат. Для компьютерных соперников должна быть реализована задержка в движении или просчетах, иначе победить их будет невозможно (особенно второго).

В данном задании предлагается реализовать приложение, которое представляет собой компьютерную игру по описанным правилам. Реализация должна включать классический вариант игры и все описанные модификации.

Визуальный интерфейс приложения представляет собой игровое поле, где происходит весь игровой процесс. Кроме того, в приложении имеется панель

настроек в виде отдельного окна, где можно задавать имя игроков, выбирать сложность игры (только для режима «игра с компьютером», влияет на подключаемого компьютерного соперника), выбирать модификацию игры. Панель настроек открывается по нажатию соответствующей кнопки, расположенной внизу игрового поля. Также внизу игрового поля должны присутствовать стандартные кнопки «Начать игру», «Рейтинг игроков» и т.п.

В рейтинге игроков ведется подсчет количества проведенных игр и побед для каждого игрока. Результат рейтинга должен сохраняться в текстовом файле и отображаться в отдельном окне при нажатии на соответствующую кнопку.

45. Игра «Домино»

Игра «Домино» представляет собой настольную логическую игру, рассчитанную на 2-4 игрока. Игра использует набор «Домино», состоящий из 28 костяшек, каждая костяшка представляет собой прямоугольную плитку (длина вдвое больше ширины) с двумя цифрами. Цифры из множества {0,1,2,3,4,5,6} наносятся в виде множества точек (как на игральных костях) на обе половины плитки во всех возможных сочетаниях. Другая сторона костяшек пуста. Костяшки с одинаковым числом очков на обоих половинах называются «дублями».

В начале игры играющим раздается по 7 костяшек перемешанных случайным образом. Не розданные костяшки (при числе игроков менее четырех) образуют «базар» и выкладываются чистой стороной к игрокам.

Первым ходит игрок, у которого имеется костяшка-«дубль» «1:1», если таковой нет, то ходит игрок, у которого есть костяшка-«дубль» «2:2» и т.п. Если дублей на руках нет (такое редко, но бывает), то начинают игру с костяшки с минимальным числом очков «0:1», «1:2» и т.п. С дубля «0:0» ходить запрещается. Игроки ходят по очереди (по часовой стрелке). Ход заключается в выкладывании костяшки на игровой стол. Следующий по очереди игрок должен выложить такую костяшку, которая образует продолжение цепочки костяшек на столе. Костяшки могут примыкать друг к другу только одинаковыми цифрами, при этом «дубли» принято выкладывать поперек образующейся цепочки. Если у игрока нет соответствующей костяшки, то он «идет на базар», забирая по одной костяшке из «базара», пока не найдет костяшку, позволяющую совершить ход. Если на «базаре» не осталось ни одной костяшки, игрок пропускает ход. Цель игры – первым избавиться от всех костяшек. При этом остальные игроки подсчитывают число очков, оставшихся у них на руках. Далее игра повторяется (начинается новый раунд), пока какой-нибудь из игроков не наберет 100 и более очков. Этот игрок объявляется проигравшим.

Возможна ситуация, когда ни один из игроков не способен совершить ход, но и на «базаре» не осталось ни одной костяшки. Такая ситуация называется «рыбой». В этом случае раунд завершается, а все игроки подсчитывают число очков на оставшихся у них на руках костяшках и добавляют к своим суммам.

Модификации классического варианта игры включают:

– «маггис» или английское домино. Первый игрок может зайти с любой костяшки, первый встретившийся в раскладке «дубль» открывает четырехсто-

роннюю цепочку. Начиная с этого момента, пристраивать новые костяшки можно со всех четырех сторон к этому «дублю»;

– «генерал». В данной разновидности игры цепочку можно закончить только «дублем» «0:0» или «6:6». Начинать игру с этих дублей запрещается.

Компьютерные соперники (которых также необходимо реализовать) могут иметь разный интеллект и разные алгоритмы игры:

– выбирает случайную костяшку, не нарушающую правил игры;

– стремится в первую очередь избавиться от «дублей» на руках;

– стремится пойти костяшкой, которая имеет на обратной стороне такое число очков, которое встречается максимальное число раз в уже выложенной цепочке (продолжить игру в таких условиях становится сложно, так как костяшки с этим числом очков будут редко встречаться, и следующий игрок будет вынужден «идти на базар».

В данном задании предлагается реализовать приложение, которое представляет собой компьютерную игру по описанным правилам в режиме «человек» – «компьютер» (т.е. все соперники от 1 до 3 – компьютерные). Реализация должна включать классический вариант игры и все описанные модификации.

Визуальный интерфейс приложения представляет собой игровое поле, где происходит весь игровой процесс (необходимо изобразить игровое поле с костяшками игроков – в перевернутом виде для компьютерных соперников, «базар» и цепочку выложенных костяшек). Кроме того, в приложении имеется панель настроек в виде отдельного окна, где можно задавать имя игрока, число соперников, выбирать сложность игры (влияет только на компьютерных соперников, например при слабом уровне игры соперники выбираются из числа тех, что выбирают случайную костяшку, а при сложном – те, что стремятся усложнить игру соперникам) и ее модификацию. Панель настроек открывается по нажатию соответствующей кнопки, расположенной внизу игрового поля. Также внизу игрового поля должны присутствовать стандартные кнопки «Начать раунд», «Взять костяшку», «Сделать ход», «Рейтинг игроков» и т.п.

В рейтинге игроков ведется подсчет количества проведенных игр и набранных игроками очков в зависимости от уровня сложности. Результат рейтинга должен сохраняться в текстовом файле и отображаться в отдельном окне при нажатии на кнопку.

46. Игра «Колбаса»

Игра «Колбаса» представляет собой настольную логическую игру, рассчитанную на 2 игрока в режиме «игра с компьютером». Игра использует набор «Домино», состоящий из 28 костяшек, каждая костяшка представляет собой прямоугольную плитку (длина вдвое больше ширины) с двумя цифрами. Цифры из множества {0,1,2,3,4,5,6} наносятся в виде множества точек (как на игральных костях) на обе половины плитки во всех возможных сочетаниях. Другая сторона костяшек пуста. Костяшки с одинаковым числом очков на обоих половинах называются «дублями».

В начале игры играющим раздается по 14 костяшек перемешанных случайным образом. Костяшки выкладываются вертикально пустой стороной

вплотную друг к другу, образуя циклические очереди для первого и второго игроков. Ход заключается в том, что игрок берет костяшку слева очереди (не показывая сопернику). Если костяшка может быть использована для текущего хода, то она выкладывается на игровой стол в открытом виде (т.е. игрок делает ход), если нет – костяшка возвращается в очередь справа (по прежнему пустой стороной вверх). Если игроку удалось сделать текущий ход, то он делает и следующий, пока у него есть такая возможность. Если текущий ход невозможен – право хода получает соперник. Цель игры – первым избавиться от всех костяшек в своей очереди.

Правила выставления костяшек заключаются в следующем. Сначала игроки (перебирая свои костяшки по описанному выше принципу) ищут дубль «1:1», с которого начинается игра. Этот дубль выкладывается вертикально слева игрового поля, после чего к нему начинают достраивать две горизонтальные цепочки костяшек, направленные вправо, формируя «колбасу». Т.е. «колбаса» представляет собой две горизонтальные и примыкающие друг к другу цепочки. Другие дубли также используются в игре, и выкладываются горизонтально. Если игроку выпадет костяшка, которую можно уложить вертикально в середине «колбасы», то игрок может сделать это («отрезать колбасу»), при этом все костяшки, образовавшиеся слева от линии отреза, переворачиваются пустой стороной вверх и добавляются в очередь сопернику. При желании игрок может не выкладывать костяшку, пропуская свой ход.

Модификации классического варианта игры включают:

– перемешивание. Несколько раз за всю игру игрок может перемешать случайным образом костяшки в своей очереди;

– один раз за всю игру игрок может «отрезать колбасу», сместив одну из цепочек влево или вправо, но не более чем на одну костяшку.

Для игры также необходимо создать компьютерных соперников, реализующих разные игровые стратегии (перечислены в порядке возрастания сложности):

– костяшки выкладываются, формируя «колбасу», но не «отрезая» ее;

– костяшки выкладываются, формируя «колбасу». Если выпавшей костяшкой можно либо «отрезать колбасу», либо положить ее в цепочку, костяшка кладется в цепочку. Если можно только «отрезать», «колбаса отрезается»;

– костяшки выкладываются, формируя «колбасу». Если выпавшей костяшкой можно либо «отрезать колбасу», либо положить ее в цепочку, «колбаса отрезается». Если можно только «отрезать», «колбаса отрезается».

В данном задании предлагается реализовать приложение, которое представляет собой компьютерную игру по описанным правилам в режиме «человек» – «компьютер». Реализация должна включать классический вариант игры и все описанные модификации.

Визуальный интерфейс приложения представляет собой игровое поле, где происходит весь игровой процесс (необходимо изобразить «колбасу», очереди обоих игроков, а также анимацию перекладывания костяшек). Кроме того, в приложении имеется панель настроек в виде отдельного окна, где можно задавать имя игрока, выбирать сложность игры (влияет только на компьютерных

соперников, например при слабом уровне игры соперник выбирается тот, что не «отрезает колбасу», а при сложном – тот, что «отрезает колбасу» в любом случае) и ее модификацию. Панель настроек открывается по нажатию соответствующей кнопки, расположенной внизу игрового поля. Также внизу игрового поля должны присутствовать стандартные кнопки «Начать игру», «Сделать ход», «Отрезать колбасу», «Рейтинг игроков» и т.п.

В рейтинге игроков ведется подсчет количества проведенных игр и набранных игроком очков в зависимости от уровня сложности. Результат рейтинга должен сохраняться в текстовом файле и отображаться в отдельном окне при нажатии на кнопку.

47. Игра «Гонки»

Игра «Гонки» представляет собой казуальную игру, рассчитанную на одного игрока. Игровое поле представляет собой прямоугольник, отображающий часть трассы. Игровой объект (машина) находится внизу поля и закреплена неподвижно, в то время как трасса движется, создавая иллюзию движения на трассе. Цель игры – как можно дольше удержаться на трассе, не врезавшись в препятствия или встречные машины. За прохождение очередного препятствия (или встречной машины) игроку начисляются очки. Встречные машины отличаются от препятствий тем, что они движутся относительно трассы в обратном направлении, в то время как препятствия неподвижны. Кроме того, на трассе встречаются бонусы, наехав на которые, игрок получает дополнительные очки.

Игрок управляет машиной, перемещая ее влево-вправо, кроме того может разгоняться до максимальной и притормаживать до минимальной скорости (остановиться полностью невозможно, поехать в обратную сторону тоже). Максимальная и минимальная скорость зависят от текущего уровня игры. После старта игры скорость плавно увеличивается до максимальной с некоторым фиксированным шагом. Игра реализуется в двумерном представлении, для упрощения допускается использовать пиксельную графику.

Уровень игры автоматически меняется, как только игрок достигнет определенного порога очков. Количество препятствий и встречных машин возрастает с увеличением уровня.

Модификации классического варианта игры включают:

– режим соревнования, в котором необходимо обогнать компьютерного соперника. В этом случае время игры ограничено, побеждает тот, кто будет лидером в гонке на момент окончания игры и не врежется в препятствие во время гонки. Соперник на трассе в данном режиме только один, но перед началом игры его можно выбрать из перечня;

– ночной режим, при котором игрок видит только часть трассы, попадающую в обзор включенных фар;

– режим «погода», при котором на трассе случайным образом генерируются лужи, наезжая на которые, машина теряет скорость, причем скорость машины (при прохождении нескольких луж подряд) может стать меньше минимальной и даже остановиться совсем. Для того, чтобы выехать из лужи (или

двигаться по сплошным лужам), игроку необходимо «газануть», стандартного ускорения машины для этого недостаточно.

Компьютерные соперники, которых следует реализовать для режима соревнования, могут действовать по следующим алгоритмам:

– перемещение влево-вправо так, чтобы машина могла гарантированно обойти все препятствия. При этом необходимо установить максимальный шаг перемещения за один такт моделирования, так как если компьютерный соперник будет перемещаться мгновенно, он никогда не врежется в препятствие и его невозможно будет победить;

– перемещение влево-вправо с торможением. Модификация предыдущего алгоритма, при этом машина притормаживает, если видно, что она не успевает объехать препятствие;

– объезд луж. По возможности, машина старается объезжать лужи, если они есть на дороге, чтобы не снижать скорости.

В данном задании предлагается реализовать приложение, которое представляет собой компьютерную игру по описанным правилам. Реализация должна включать классический вариант игры и все описанные модификации.

Визуальный интерфейс приложения представляет собой игровое поле, где происходит весь игровой процесс (необходимо изобразить трассу, машины и препятствия на ней). Кроме того, в приложении имеется панель настроек в виде отдельного окна, где можно задавать имя игрока, выбирать сложность игры (влияет на максимальную скорость перемещения трассы, на алгоритм компьютерного соперника в режиме соревнования, а также на максимальный шаг перемещения компьютерного соперника, так как чем он выше, тем меньше вероятность того, что соперник врежется в препятствие). Панель настроек открывается по нажатию соответствующей кнопки, расположенной внизу игрового поля. Также внизу игрового поля должны присутствовать стандартные кнопки «Начать игру», «Рейтинг игроков» и т.п.

В рейтинге игроков ведется подсчет набранных игроком очков в зависимости от уровня сложности. Результат рейтинга должен сохраняться в текстовом файле и отображаться в отдельном окне при нажатии на кнопку.

48. Игра «Бомбер»

Игра «Бомбер» представляет собой казуальную игру, рассчитанную на одного игрока. Игра происходит на прямоугольном поле, разделенном на квадратные ячейки и включает несколько уровней. Каждый уровень представляет собой лабиринт, составленный из кирпичных и бетонных стен. Игрок управляет «бомбером» – игровым персонажем, который может перемещаться по лабиринту в четырех направлениях. «Бомбер» может перемещаться только по свободным ячейкам игрового поля и не может проходить сквозь стены. Кроме того, на игровом поле имеется несколько неподвижных целей, которые должен взорвать «бомбер». Для этого «бомбер» может заминировать текущую ячейку игрового поля – ту самую, в которой находится. После этого бомба автоматически взрывается спустя несколько секунд, поражая объекты в радиусе нескольких ячеек игрового поля. Этими объектами являются неподвижные цели, кирпичные (но

не бетонные) стены. Если «бомбер» не успеет отойти от бомбы на безопасное расстояние за отведенное время, он погибает. При взрыве каждой цели игроку начисляются очки. Если игрок смог уничтожить несколько целей одной бомбой, число начисляемых очков увеличивается (режим «комбо»). Игрок выигрывает текущий уровень, если ему удалось взорвать все цели.

Кроме того, на поле имеется несколько противников, которые могут перемещаться (также по четырем направлениям). Если такой противник попадет в радиус действия бомбы в момент взрыва, он погибает. Если же «бомбер» столкнется с противником, то погибает уже он.

Модификации игры включают:

– наличие в инвентаре у «бомбера» бомб разного радиуса действия и с разной задержкой срабатывания;

– наличие в инвентаре у «бомбера» направленных бомб, которые могут разрушать объекты только в одном направлении;

– наличие в инвентаре у «бомбера» дистанционных бомб, которые можно подорвать при помощи дистанционного взрывателя, отойдя на безопасное расстояние;

Выбор вида бомбы осуществляется по нажатию специальной кнопки. «Бомбер» всегда имеет неограниченное количество простых бомб и ограниченное количество модифицированных.

Компьютерные противники, которые имеются на уровне, могут действовать по следующим алгоритмам:

– перемещение по четырем направлениям в случайном порядке (разумеется не наезжая на стены);

– перемещение по одному выбранному направлению до столкновения со стеной или границей игрового поля, после чего новое направление перемещения выбирается случайно;

– перемещение к «бомберу» с целью преследования, однако если на пути имеется непроходимая стена, то преследование прекращается (реализовывать обход стен не нужно);

Во всех случаях компьютерные противники также могут сбрасывать бомбы с некоторой вероятностью, но только самые простые, без модификаций. Разумеется, если компьютерный противник или «бомбер» попадет в радиус действия бомбы в момент взрыва, он погибнет, так что компьютерные противники могут подорвать сами себя.

В данном задании предлагается реализовать приложение, которое представляет собой компьютерную игру по описанным правилам. Реализация должна включать классический вариант игры и все описанные модификации. Для игры необходимо разработать минимум 10 уровней для демонстрации работы приложения. Уровни могут храниться в текстовых файлах (формат хранения придумать самостоятельно).

Визуальный интерфейс приложения представляет собой игровое поле, где происходит весь игровой процесс (необходимо изобразить уровень, перемещение игровых персонажей, стены, бомбы и инвентарь «бомбера»). Кроме того, в приложении имеется панель настроек в виде отдельного окна, где можно зада-

вать имя игрока, выбирать сложность игры (влияет на количество компьютерных противников на поле и выбираемый алгоритм их действия). Панель настроек открывается по нажатию соответствующей кнопки, расположенной внизу игрового поля. Также внизу игрового поля должны присутствовать стандартные кнопки «Начать игру», «Рейтинг игроков» и т.п.

В рейтинге игроков ведется подсчет набранных игроком очков в зависимости от уровня сложности. Результат рейтинга должен сохраняться в текстовом файле и отображаться в отдельном окне при нажатии на кнопку.

49. Игра «Крестики-нолики – 5 в ряд»

Игра «Крестики-нолики – 5 в ряд» представляет собой казуальную логическую игру, рассчитанную на одного или двух игроков (режим «игры с компьютером» или «друг против друга»). В последнем случае игроки играют друг против друга по очереди за одним компьютером. Игровое поле представляет собой квадратную область, разбитую на ячейки. В каждой ячейке может быть нарисован либо крестик, либо нолик. Первый игрок использует крестики, второй – нолики. Игроки ходят по очереди.

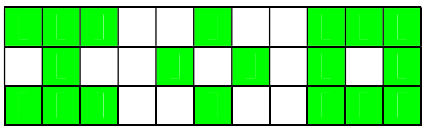
В начале игры разыгрывается правило первенства хода любым способом, например, при помощи игральной кости, которую следует бросить по очереди обоим игрокам. Игрок, который выбросит больше очков, начинает игру (и ставит крестики), его соперник играет вторым (и ставит нолики).

Цель игры заключается в выстраивании непрерывной линии из пяти крестиков или пяти ноликов, причем линия может быть горизонтальной, вертикальной, либо диагональной, но кратной 45°. Игрок, который сможет построить такую линию первым, становится победителем. Игра может закончиться и ничьей, если игровое поле окажется полностью заполненным, но построить линию так и не удалось.

Модификации классического варианта игры включают:

– нестандартная форма поля (например в виде креста или зигзага, что придает дополнительный интерес игре);

– иное условие победы (например, вместо линии для победы необходимо выстроить зигзаг, окружность или квадрат) – см. рис.;



– наличие на поле отверстий, в которые нельзя ставить ни крестики, ни нолики;

– наличие на поле специальных ячеек, которые игрок может занять лишь тогда, когда рядом с этой ячейкой будет выставлен либо крестик либо нолик. Разумеется, ячейку может занять только тот игрок, который выставит рядом с ней свой знак (крестик или нолик) первым.

В данном задании предлагается реализовать приложение, которое представляет собой компьютерную игру по описанным правилам в режиме «игра с компьютером» или «друг против друга». Реализация должна включать класси-

ческий вариант игры и все описанные модификации. Для игры необходимо разработать минимум пять игровых полей и сохранить их в текстовом файле (формат хранения придумать самостоятельно). Максимальный размер поля – 100х100 ячеек. В режиме «игра с компьютером» компьютер должен самостоятельно вычислять свой следующий ход из числа допустимых вариантов и реализовывать некоторую выигрышную стратегию.

Визуальный интерфейс приложения представляет собой игровое поле, где происходит весь игровой процесс (необходимо изобразить игровое поле, кость для розыгрыша первого хода, игровой таймер, количество сделанных ходов каждым из игроков). Кроме того, в приложении имеется панель настроек в виде отдельного окна, где можно задавать имя игрока, выбирать игровое поле, задавать условие победы. Панель настроек открывается по нажатию соответствующей кнопки, расположенной внизу игрового поля. Также внизу игрового поля должны присутствовать стандартные кнопки «Начать игру», «Рейтинг игроков» и т.п.

В рейтинге игроков ведется подсчет количества проведенных игр, набранных очков и времени, затраченного на игру. Результат рейтинга должен сохраняться в текстовом файле и отображаться в отдельном окне при нажатии на соответствующую кнопку.

50. Игра «Танки»

Игра «Танки» представляет собой казуальную компьютерную игру, рассчитанную на одного игрока. Игровое поле – прямоугольное, разбитое на квадратные ячейки. Игра включает несколько уровней. Цель игры заключается в прохождении всех уровней, для этого надо выполнить цели каждого уровня. ПО мере выполнения целей уровня игроку начисляются очки.

Каждый уровень представляет собой лабиринт, образованный кирпичными и бетонными стенами. Игрок управляет танком, который может перемещаться в четырех направлениях. Также на игровом поле имеется выход, но для того, чтобы открыть его, необходимо собрать несколько ключей. Ключи появляются в разных местах игрового поля и на строго определенное время. Если игрок не успеет взять ключ, то он исчезает и появляется в другом месте. Ключи появляются в случайных местах (но не на стенах), количество ключей зависит от сложности уровня.

Помимо этого на уровне имеются снаряды и ремкомплекты, которые можно подбирать. Ремкомплект добавляет 75% жизней танку. Если танк имеет 100% жизней, то ремкомплект не подбирается и не расходуется.

Танк может стрелять снарядами трех типов. Снаряды первого типа могут разрушать лишь кирпичные стены уровня, снаряды второго типа – кирпичные и бетонные стены, снаряды третьего типа также разрушают кирпичные и бетонные стены, но их радиус действия составляет 3 клетки. Количество снарядов первого типа бесконечно, остальные снаряды необходимо подбирать.

На игровом поле также присутствуют компьютерные соперники. Они также могут стрелять, но только снарядами первого типа, снаряды второго и третьего типа они не подбирают, даже если случайно наедут на них. Также

компьютерные соперники не подбирают ремкомплекты. Т.о. компьютерные соперники способны пробивать только кирпичные стены на игровом поле.

Помимо пробивания стен снарядом можно поразить танк игрока или компьютерного соперника, так как каждый снаряд отнимает некоторое количество жизней. Снаряды первого типа отнимают 15% жизней, второго – 25% жизней, третьего – 40% жизней.

Для того чтобы выиграть уровень, необходимо собрать все ключи и уничтожить всех соперников.

Модификации игры включают:

– нестандартная форма поля (например в виде креста или зигзага, что придает дополнительный интерес игре);

– ночной режим, при котором видимость вокруг танка составляет шесть игровых клеток;

– режим разведчика, при котором видимость вокруг танка составляет четыре игровые клетки, но ранее посещенные участки игрового уровня остаются видимыми, даже если танк игрока удаляется от них;

Компьютерные соперники, которые имеются на уровне, могут действовать по следующим алгоритмам:

– перемещение по четырем направлениям в случайном порядке (разумеется не наезжая на стены);

– перемещение по одному выбранному направлению до столкновения со стеной или границей игрового поля, после чего новое направление перемещения выбирается случайно;

– перемещение к танку игрока с целью преследования, однако если на пути имеется непроходимая стена, то преследование прекращается (реализовывать обход стен не нужно).

В данном задании предлагается реализовать приложение, которое представляет собой компьютерную игру по описанным правилам. Реализация должна включать классический вариант игры и все описанные модификации. Для игры необходимо разработать минимум 10 уровней для демонстрации работы приложения. Уровни могут храниться в текстовых файлах (формат хранения придумать самостоятельно).

Визуальный интерфейс приложения представляет собой игровое поле, где происходит весь игровой процесс (необходимо изобразить уровень, перемещение игровых персонажей, стены, подбираемые элементы, показать число набранных очков, число жизней и снарядов каждого типа). Кроме того, в приложении имеется панель настроек в виде отдельного окна, где можно задавать имя игрока, выбирать сложность игры (влияет на количество ключей, количество компьютерных противников на поле и выбираемый алгоритм их действия). Панель настроек открывается по нажатию соответствующей кнопки, расположенной внизу игрового поля. Также внизу игрового поля должны присутствовать стандартные кнопки «Начать игру», «Рейтинг игроков» и т.п.

В рейтинге игроков ведется подсчет набранных игроком очков в зависимости от уровня сложности. Результат рейтинга должен сохраняться в текстовом файле и отображаться в отдельном окне при нажатии на кнопку.

Заключение

В методических указаниях приводятся рекомендации по выполнению курсового проекта по дисциплине «Объектно-ориентированное программирование» для студентов, обучающихся по направлениям подготовки 02.03.01 «Математика и компьютерные науки», 09.03.02 «Информационные системы и технологии» всех форм обучения.

В рамках выполнения курсового проекта обучающимся предлагается разработать приложение на языке C++ одного из трех типов:

– специализированное приложение, предназначенное для редактирования данных указанного вида или для выполнения специализированных расчетов;

– моделирующее приложение, позволяющее сымитировать работу реального технологического процесса на заводе или в бытовой ситуации;

– игровое приложение, реализующее правила настольной логической или казуальной игры.

Методические указания включают перечень основных требований, которые обучающиеся должны учесть при разработке приложения. Большая часть из этих требований сложилась, исходя из стандартов разработки приложений профессионального уровня.

Вместе с тем, следует отметить, что методика разработки алгоритмов, а также синтаксис языка программирования C++ в настоящих указаниях подробно не рассматривается. Это объясняется тем, что вся эта информация либо изучалась в предыдущих дисциплинах, либо имеется в методических указаниях к выполнению лабораторных работ и конспекте лекций.

В методических указаниях также рассматривается структура пояснительной записки (из каких структурных элементов она состоит), а также информация о том, как правильно должны быть оформлены основные блоки, входящие в эти элементы (рисунки, таблицы, списки, листинги программного кода). Данная информация может оказаться полезной не только при оформлении пояснительной записки для данной работы, но и для последующих курсовых проектов или работ, а также дипломного проектирования. Также описывается примерное содержание пояснительной записки (какие основные разделы и приложения должна содержать пояснительная записка, и что в них должно быть описано).

В заключение кратко рассматривается процедура защиты курсового проекта и приводятся критерии, по которым оценивается выполненная работа.

Методические указания включают 50 индивидуальных вариантов заданий. Каждое из заданий имеет подробное описание приложения, которое необходимо разработать обучающемуся и должным образом формализовано. Так, даже для известных большинству людей игр имеется строгое описание правил. Для выполнения курсового проекта обучающийся вправе выбрать любое из заданий по своему усмотрению. Все предлагаемые задания имеют примерно одинаковый уровень сложности, рассчитанный на обучающихся программированию студентов уровня подготовки выше среднего.

Список рекомендуемых источников

1. Павловская Т. А. C++. Объектно-ориентированное программирование: Практикум [Текст] / Т. А. Павловская, Ю. А. Щупак // СПб.: Питер, 2006. — 265 с: ил.

2. Шилдт Г. C++: руководство для начинающих [Текст] / Герберт Шилдт // Пер. с англ. — М.: Изд. дом «Вильямс», 2005. — 2-е издание. — 672 с. : ил. — Парал. тит. англ.

3. Шилдт Г. Искусство программирования на C++ [Текст] / Герберт Шилдт // СПб.: БХВ Петербург, 2005. — 496 с. : ил.

4. Лафоре Р. Объектно-ориентированное программирование в C++. Классика Computer Science [Текст] / Роберт Лафоре // СПб.: Питер, 2022. — 4-е издание. — 928 с.: ил. — (Серия «Классика computer science»).

5. Фримен Э. Head First: Паттерны проектирования [Текст] / Эрик Фримен, Элизабет Робсон // Пер. с англ. — СПб.: Питер, 2018. —656 с.: ил.

6. TX Library Документация [Электронный ресурс] / сост. Илья Дединский // URL: http://storage.ded32.net.ru/Lib/TX/TXUpdate/Doc/HTML.ru/ (дата обращения 01.02.2022).

ПРИЛОЖЕНИЕ А
Образец титульного листа пояснительной записки

ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
«ДОНЕЦКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовому проекту
по дисциплине «Объектно-ориентированное программирование»
на тему: «Объектно-ориентированная реализация программного приложения»

Руководители:
доцент кафедры КМД
Павлий В.А.
ассистент кафедры КМД
Бабакина А.А.

Выполнил:
студент группы МИД-21б
Иванов И.И.

Донецк, 2022

ПРИЛОЖЕНИЕ Б
Бланк технического задания
(печатать с двух сторон, желтые фрагменты изменить под свои данные)

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ ДНР
ДОНЕЦКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Факультет Информационные системы и технологии
Кафедра Компьютерное моделирование и дизайн
Специальность Информационные системы и технологии

УТВЕРЖДАЮ:
зав. кафедрой КМД
Карабчевский В.В.
“___” _____ 20__ г.

Техническое задание
на курсовой проект
по курсу “Объектно-ориентированное программирование”
студента группы _____
_____
(фамилия, имя, отчество)

Тема работы: Объектно-ориентированная реализация программного приложения “_____”.

Цель работы: Практическое закрепление знаний по составлению алгоритмов и разработке объектно-ориентированного приложений на языке C++ профессионального уровня, а также оформлению сопроводительной документации.

1. Срок защиты курсового проекта “___” _____ 20__ г.
2. Требования к разработке программного продукта:
   – язык программирования – C++;
   – среда разработки – Visual Studio или ее аналоги;
   – использование библиотек C++ для работы с графикой и окнами;
   – использование паттернов проектирования (минимум один паттерн);
   – модульная структура проекта (минимум три модуля);
   – визуализация (если нужно, то и анимация) в графическом режиме.
3. Содержание пояснительной записки:
   – титульный лист;
   – реферат;
   – введение;
   – постановка задачи;
   – выбор инструментария для разработки;
   – проектирование программного продукта;
   – объектно-ориентированный анализ программного продукта;
   – разработка программного продукта;
   – описание программного продукта;
   – заключение;
   – список использованных источников;

– приложение А. Техническое задание;
– приложение Б. Листинг программы;
– приложение В. Графическая часть проекта;
– приложение Г. Экранные формы.

4. Условие задания (вариант № ___)

Скопировать полный текст задания из варианта

5. Календарный план выполнения работы:

№ этапа	Наименование этапа	Срок выполнения
1	Выдача задания, составление ТЗ и его утверждение	2 неделя
2	Проектирование программного продукта	3 неделя
3	Определение структур данных	4 неделя
4	Разработка основных алгоритмов	7 неделя
5	Написание программы	12 неделя
6	Отладка программы	14 неделя
7	Разработка пояснительной записки	16 неделя
8	Защита курсовой работы	17 неделя

Дата выдачи задания “___” _____ 20__ г.

Задание принял(а) к исполнению студент _____ _____
(подпись) (ФИО)

Руководители: _____ _____ В. А. Павлий
(дата) (подпись)

_____ _____ _____
(дата) (подпись) (ФИО)

_____ _____ _____
(дата) (подпись) (ФИО)

ПРИЛОЖЕНИЕ В
Образец реферата

Реферат

Страниц: 45, рисунков: 14, таблиц: 2, приложений: 5, источников: 6

Тема работы: Объектно-ориентированная реализация программного приложения «Запомни числа».

Цель работы: Практическое закрепление знаний по составлению алгоритмов и разработке объектно-ориентированного приложений на языке C++ профессионального уровня, а также оформлению сопроводительной документации.

Объектом разработки данного курсового проекта является программное приложение с графическими и интерактивными элементами, которое имеет название «Запомни числа».

Среда разработки: программа была разработана в интегрированной среде разработки Microsoft Visual Studio 2019.

В ходе данной работы было разработано игровое приложение «Запомни числа», рассчитанное на двух игроков. Приложение написано на объектно-ориентированном языке программирования C++ с использованием классов и паттернов, а также кроссплатформенной мультимедийной библиотеки TX Library, содержащей ряд модулей для создания простых графических приложений.

ООП, ПАТТЕРН ПРОЕКТРИРОВАНИЯ, КЛАСС, ОБЪЕКТ,
МОДУЛЬ ПРИЛОЖЕНИЯ, TX LIBRARY, ДИАГРАММА КЛАССОВ

					09.03.02.КП2022.18\6861.00.00 ПЗ			
Изм.	Лист	№ документа	Подпись	Дата				
Студент		Иванов И.И.			Разработка персонального информационного сайта. Пояснительная записка	Лит	Лист	Листов
Руковод.		Павлий В.А.					1	1
Руковод.		Бабакина А.А.				ДонНТУ, ФИСТ, кафедра КМД, гр.МИД18-б		
Н.контр.		Павлий В.А.						

ПРИЛОЖЕНИЕ Г
Примеры библиографических описаний

Книги без автора

Политология : учеб. пособие / сост. А. Иванов. – СПб. : Высш. школа, 2003. – 250 с.

Основы политологии : словарь / под ред. А. Г. Белова, П. А. Семина. – М. : Мысль, 2005. – 350 с.

Малый бизнес : перспективы развития : сб. ст. / под ред. В. С. Ажаева. – М. : ИНИОН, 1991. – 147 с.

Книги одного автора

Игнатов, В. Г. Государственная служба субъектов РФ : Опыт сравнительно-правового анализа : науч.– практ. пособие / В. Г. Игнатов. – Ростов н/Д : СКАГС, 2000. – 319 с.

Базаров, Т. Ю. Управление персоналом : учеб. пособие / Т. Ю. Базаров. – М. : Академия, 2003. – 218 с.

Балабанов, И. Т. Валютные операции / И.Т. Балабанов. – М. : Финансы и статистика, 1993. – 144 с.

Книги двух авторов

Корнелиус, Х. Выиграть может каждый : Как разрешать конфликты / Х. Корнелиус, З. Фэйр ; пер. П. Е. Патрушева. – М. : Стрингер, 1992. – 116 с.

Смирнов, К. Высшая математика : учебник / К. Смирнов, В. Петров. – М. : Университет, 2003. – 220 с.

Агафонова, Н. Н. Гражданское право : учеб. пособие / Н. Н. Агафонова, Т. В. Богачева ; под общ. ред. А. Г. Калпина. – М. : Юрист, 2002. – 542 с.

Ершов, А. Д. Информационное управление в таможенной системе / А. Д. Ершов, П. С. Конопаева. – СПб. : Знание, 2002. – 232 с.

Игнатов, В. Г. Профессиональная культура и профессионализм государственной службы : контекст истории и современность / В. Г. Игнатов, В. К. Белолипецкий. – Ростов н/Д : МарТ, 2000. – 252 с.

Книги трех авторов

Киселев, В.В. Анализ научного потенциала / В. В. Киселев, Т. Е. Кузнецова, З. З. Кузнецов. – М. : Наука, 1991. – 126 с.

Громов, С. Экономика : сб. ст. / С. Громов, Н. Тихонов, Т. Глушкова. – М. : ЭКСМО, 2001. – 230 с.

Журавлев, П. В. Мировой опыт в управлении персоналом : обзор зарубежных источников / П. В. Журавлев, М. Н. Кулапов, С. А. Сухарев. – М. : Рос. Экон. Акад. ; Екатеринбург : Деловая книга, 1998. – 232 с.

Аяцков, Д. Ф. Кадровый потенциал органов местного самоуправления : проблемы и опыт оценки / Д. Ф. Аяцков, С. Ю. Наумов, Е. Н. Суетенков. – Саратов : ПАГС, 2001. – 135 с.

Книги четырех и более авторов

Управленческая деятельность : структура, функции, навыки персонала / К. Д. Скрипник [и др.]. – М. : Приор, 1999. – 189 с.

Философия : университетский курс : учебник / С. А. Лебедев [и др.] ; под общ. ред. С. А. Лебедева. – М. : Гранд, 2003. – 525 с.

Управление персоналом : от фактов к возможностям будущего : учеб. пособие / А. А. Брасс [и др.] – Минск : УП «Технопринт», 2002. – 387 с.

Словари и энциклопедии

Социальная философия : словарь / под общ. ред. В. Е. Кемерова, Т. Х. Керимова. – М. : Академический Проект, 2003. – 588 с.

Ожегов, С. И. Толковый словарь русского языка / С. И. Ожегов, Н. Ю. Шведова. – М. : Азбуковник, 2000. – 940 с.

Чернышев, В. Н. Подготовка персонала : словарь / В. Н. Чернышев, А. П. Двинин. – СПб. : Энергоатомиздат, 2000. – 143 с.

Экономическая энциклопедия / Е. И. Александрова [и др.]. – М. : Экономика, 1999. – 1055 с.

Многотомные издания

История дипломатии : В 5 т. Т. 5. / под ред. А. А. Громыко. – М. : Госполитиздат, 1959. – 766 с.

Официальные документы

Конституция Российской Федерации : офиц. текст. – М. : ОСЬ-89, 2000. 48 с.

Об исполнении федерального бюджета за 2003 год : федер. закон от 4 апреля 2005 № 30-ФЗ // Собрание законодательства РФ. – 2005. – № 15. Ст. 1275.

О системе и структуре федеральных органов исполнительной власти : указ Президента РФ от 9 марта 2004 № 314 // Собрание законодательства РФ. – 2004. – № 11. – Ст. 945.

Об инвестиционном фонде Российской Федерации : постановление Правительства от 23 ноября 2005 № 694 // Собрание законодательства РФ. – 2005. – № 48. – Ст. 5043.

Статья, раздел, глава

Бакаева, О. Ю. Таможенные органы Российской Федерации как субъекты таможенного права / О. Ю. Бакаева, Г. В. Матвиенко // Таможенное право. – М. : Юрист, 2003. – С. 51-91

Веснин, В. Р. Конфликты в системе управления персоналом / В. Р. Веснин, С. Иванов // Практический менеджмент персонала. – М. : Юрист, 1998. – С. 395-414

Иванов, С. Проблемы регионального реформирования // Экономические реформы / под ред. А. Е. Когут. – СПб. : Наука, 1993. – С. 79-82

Из словаря

Межличностные отношения // Управление персоналом : энциклопедический словарь / под ред. А. Я. Кибанова [и др.]. – М. : ИНФРА-М, 1998. – С. 240 - 241.

Руднев, В. П. Модерн в искусстве / В. П. Руднев // Словарь культуры XX века: ключевые понятия и тексты. – М. : Аграф, 1999. – С.119-124.

Статьи из газет

Титов В. Банковская система Северо-Запада России / В. Титов // Экономика и жизнь. – 2005. – № 1. – С. 38.

Серов А. Итоги национализации / А. Серов // Известия. – 2000. – 14 июня. – С. 5.

Статьи из журналов

Терентьева Т. Банковские услуги : спрос и предложение / Т. Терентьева // Деньги и кредит . – 2005. – №. 12. – С. 54-57.

Беков Т. Конституционные конфликты / Т. Беков // Государство и право. – 2004. – № 11. – С.19-25

Роль права в обеспечении интересов в Федерации // Журнал российского права. – 2005. – №. 12. – С. 141-146

Электронные ресурсы локального доступа
(с информацией, зафиксированной на отдельном физическом носителе)

Большая энциклопедия Кирилла и Мефодия 2000 [Электронный ресурс]. – М.: Кирилл и Мефодий, 2000. – 2 электрон. опт. диск

Электронные ресурсы удаленного доступа
(представленные в Интернете или внутренних сетях)

Руководство : как создавать контент и писать тексты для веб-сайтов? [Электронный ресурс]. – Режим доступа : http://arcobaleno-ru.livejournal.com / 16328 .html.

МЕТОДИЧЕСКИЕ УКАЗАНИЯ
к выполнению курсового проекта
по дисциплине «Объектно-ориентированное программирование»

Составитель:

Павлий Виталий Александрович – кандидат технических наук, доцент кафедры компьютерного моделирования и дизайна ГОУВПО «ДОННТУ»

Ответственный за выпуск:

Карабчевский Виталий Владиславович – кандидат технических наук, доцент, заведующий кафедрой компьютерного моделирования и дизайна ГОУВПО «ДОННТУ»