Министерство образования Республики Беларусь
Учреждение образования
«Белорусский государственный университет
информатики и радиоэлектроники»

Кафедра электронных вычислительных машин

КОНСТРУИРОВАНИЕ ПРОГРАММ И ЯЗЫКИ ПРОГРАММИРОВАНИЯ

МЕТОДИЧЕСКИЕ УКАЗАНИЯ
ПО КУРСОВОМУ ПРОЕКТИРОВАНИЮ
для студентов специальности
1-40 02 01 «Вычислительные машины, системы и сети»
всех форм обучения

Минск БГУИР 2010

УДК 004.421+004.43(075.8)
ББК 32.973.26-018.2 я 73
К65

С о с т а в и т е л и:
А. В. Бушкевич, А. М. Ковальчук, И. В. Лукьянова

Конструирование программ и языки программирования: метод. К65 указания по курсовому проектированию для студ. спец. 1-40 02 01 «Вычислительные машины, системы и сети» всех форм обуч. / сост. А. В. Бушкевич, А. М. Ковальчук, И. В. Лукьянова. – Минск : БГУИР, 2010. – 30 с. : ил.

В методических указаниях излагаются цели и задачи курсовой работы, требования к содержанию, объему и оформлению пояснительной записки.

В первых четырех разделах методических указаний рассматриваются общие вопросы по содержанию и оформлению пояснительной записки, а также вопросы по построению и оформлению блок-схем алгоритмов и диаграммы классов. В пятом разделе приводятся варианты заданий. Методические указания предназначены для студентов, выполняющих курсовую работу по дисциплине «Конструирование программ и языки программирования».

УДК 004.421+004.43 (075.8)
ББК 32.973.26-018.2 я 73

СОДЕРЖАНИЕ

1 ЦЕЛИ И ЗАДАЧА КУРСОВОЙ РАБОТЫ

Цели курсовой работы: овладеть практическими навыками проектирования и разработки законченного, отлаженного и протестированного программного продукта с использованием языка высокого уровня C++ и машинно-ориентированного языка Assembler; закрепить и углубить теоретические знания, полученные при изучении курсов «Основы алгоритмизации и программирования», «Введение в специальность», «Конструирование программ и языки программирования».

Задача курсовой работы – проектирование прикладного программного обеспечения: реализация игр, графических и текстовых редакторов, программ для создания, обработки и хранения данных.

2 ТРЕБОВАНИЯ К СОДЕРЖАНИЮ И ОФОРМЛЕНИЮ ПОЯСНИТЕЛЬНОЙ ЗАПИСКИ

Результатом выполнения курсовой работы по курсу «Конструирование программ и языки программирования» является разработанная и отлаженная программа, полностью отвечающая поставленным требованиям, предъявленным в разделе *«Задание на курсовую работу»*, и пояснительная записка к данной программе.

2.1 Требования к содержанию пояснительной записки

Содержание пояснительной записки должно включать следующие разделы:

– задание на курсовую работу;

– содержание с указанием страниц расположения отдельных частей пояснительной записки;

– перечень используемых сокращений;

– введение;

– обзор методов и алгоритмов решения поставленной задачи;

– обоснование выбранных методов и алгоритмов;

– описание программы для программиста;

– описание алгоритмов решения задачи;

– руководство пользователя;

– заключение;

– литература;

– приложение А (листинг программы с комментариями);

– приложение Б (скриншоты работы программы).

Рассмотрим подробнее каждый из разделов пояснительной записки.

В *«Задании на курсовую работу»* должен быть приведен текст задания, отражены рекомендации по использованию среды разработки, указаны ограничения на используемую операционную систему, а также при необходимости ограничения на значения специфических параметров, используемых в задаче. Объем данного раздела – одна страница.

«Содержание» должно включать все приведенные выше разделы. Изменение состава *«Содержания»* допускается только по согласованию с преподавателем, осуществляющим руководство выполнением курсовой работы. Объем – одна страница.

«Введение» должно содержать описание современных возможностей по разработке прикладного программного обеспечения с использованием языка высокого уровня C++ и машинно-ориентированного языка Assembler. Объем – одна – две страницы.

Раздел *«Обзор методов и алгоритмов решения поставленной задачи»* должен описывать максимально возможное число методов и алгоритмов решения поставленной задачи или отдельных ее частей с указанием их особенностей, достоинств и недостатков с точки зрения применимости для решения рассматриваемой задачи. В данном разделе в обязательном порядке должны при-

сутствовать ссылки на используемую литературу. Объем данного раздела – три – шесть страниц.

В *«Обосновании выбранных методов и алгоритмов»* описывается выбор тех или иных методов и алгоритмов решения поставленной задачи или отдельных ее частей на основании проведенного анализа в предыдущем разделе. При этом необходимо учесть сложность программирования алгоритма, точность метода, время выполнения программы, объем программы. Объем данного раздела – две – три страницы.

В разделе *«Описание программы для программиста»* описывается общая концепция построения программы, приводятся диаграмма классов (см. рекомендации в разделе 4) и ее описание, а также приводится схема данных. В описании схемы данных необходимо привести структуру файлов, их взаимосвязь, а также структуру динамических списков, например контейнеров и т. п.

В разделе *«Описание алгоритмов решения задачи»* описываются наиболее значимые алгоритмы с использованием блок-схем алгоритмов (см. рекомендации в разделе 3) и пошагового описания алгоритмов. При пошаговом описании алгоритма разные этапы алгоритма обозначаются шагами. Записывается слово «Шаг» и далее указывается порядковый номер шага. Затем необходимо на русском языке написать, что надо выполнить на данном шаге и указать имя объекта, над которым производится действие. Если после выполнения действия переход должен быть выполнен не к следующему шагу за текущим, то необходимо указать номер шага, к которому должен быть выполнен переход. Количество описываемых алгоритмов порядка 10, при этом половина описывается при помощи блок-схем алгоритмов, а половина – при помощи пошагового описания алгоритмов.

В разделе *«Руководство пользователя»* приводится перечень и назначение всех файлов, входящих в состав разработанного программного приложения, необходимых для функционирования программы. Приводится описание назначения и возможностей программы. Детально описываются все пункты меню

программы и при необходимости последовательность действий по работе с программой.

В *«Заключении»* должны быть отражены результаты выполнения курсовой работы, указано, какие новые знания были получены в ходе выполнения работы, дана характеристика разработанной программы с указанием ее достоинств и путей дальнейшего совершенствования.

В *«Литературе»* должен быть приведен список используемых при выполнении курсовой работы источников, а именно перечень научно-технических публикаций, нормативно-технических документов и других материалов.

2.2 Требования к оформлению пояснительной записки

Пояснительная записка должна быть оформлена на листах формата А4 (для чертежей допускается использование формата А3), сброшюрованных в папку. Страницы в курсовой работе должны быть пронумерованы. Нумерация страниц в работе должна быть сквозная, начиная с титульного листа и оканчивая приложениями, номера проставляются только на страницах, расположенных после содержания. Номер страницы должен располагаться в правом нижнем углу листа. Допускается не печатать номера страниц, а надписывать черной ручкой. На титульном листе приводится название учебного учреждения, факультета, кафедры, наименование дисциплины, темы курсовой работы, номер курса, группы, фамилия, имя, отчество студента и преподавателя, осуществляющего руководство выполнением курсовой работы.

Текст пояснительной записки разбивается на разделы, подразделы, пункты и подпункты. Раздел – первая ступень деления, обозначенная номером и снабженная заголовком. Подраздел – часть раздела, обозначенная номером и имеющая заголовок. Пункт – часть раздела, обозначенная номером, может иметь заголовок. Подпункт – часть пункта, обозначенная номером, может иметь заголовок. Абзац – логическое выделение части текста, не имеющее номера.

Перечень разделов пояснительной записки приведен в подразделе 2.1. Допускается помещать текст между заголовками раздела и подраздела (пункта).

Разделы должны иметь порядковые номера в пределах всей пояснительной записки, обозначаемые арабскими цифрами без точки и записанные с абзацного отступа. Номера разделов, подразделов, пунктов и подпунктов следует выделять полужирным шрифтом. Заголовки разделов пишутся прописными буквами полужирным шрифтом размером 14 – 16 пунктов. Каждый раздел должен начинаться с нового листа. Переносы слов в заголовке не допускаются, точка в конце заголовка не ставится. Заголовки подразделов записываются строчными буквами (кроме первой) полужирным шрифтом размером 14 пунктов. Цифровой индекс подраздела должен состоять из порядкового номера раздела и отделенного от него точкой порядкового номера подраздела. Номер подраздела записывается с абзацного отступа и точка после номера не ставится. Пункты нумеруют в пределах подраздела. Цифровой индекс пункта должен состоять из номера раздела, подраздела и пункта, разделенных точками, и записан с абзацного отступа. Переносы слов в заголовках не допускаются, если заголовок состоит из двух предложений, их разделяют точкой. В случае когда заголовки раздела или подраздела занимают несколько строк, вторая и последующие строки выравниваются по первой букве первой строки. Расстояние между заголовком и последующим (предыдущим) текстом, а также между заголовком раздела и подраздела должно быть равно одной строке (интервал полуторный). Не допускается помещать заголовок на одной странице, а начало текста на другой.

Разделы СОДЕРЖАНИЕ, ПЕРЕЧЕНЬ ИСПОЛЬЗУЕМЫХ СОКРАЩЕНИЙ, ВВЕДЕНИЕ, ЗАКЛЮЧЕНИЕ, ЛИТЕРАТУРА записываются в виде заголовка прописными буквами и располагаются симметрично тексту.

Термины и определения должны быть едиными и соответствовать установленным стандартам, а при отсутствии стандартов – общепринятым в научно-технической литературе. Не допускается использование нескольких синонимов для одного и того же понятия.

Сокращения слов в тексте и надписях не допускаются, за исключением сокращений, установленных ГОСТом и общепринятых в русском языке; сокращений, применяемых для обозначения программ, их частей и режимов работы в языках программирования.

Если в записке необходимо использовать нестандартное сокращение, оно должно быть пояснено при первом употреблении. При наличии значительного количества таких сокращений необходимо в начале записки, после содержания, помещать раздел ПЕРЕЧЕНЬ УСЛОВНЫХ ОБОЗНАЧЕНИЙ.

Иллюстрации могут располагаться в основном тексте или в приложениях. Иллюстрации, если их более одной, нумеруются в пределах раздела арабскими цифрами. При этом номер рисунка состоит из номера раздела и порядкового номера рисунка в разделе (2.1, 2.2 и т.д.). Иллюстрация может состоять из собственно рисунка, тематического заголовка, поясняющих данных и номера. Каждый рисунок сопровождается подрисуночной подписью, которую располагают симметрично полю, занимаемому иллюстрацией. Подпись должна содержать слово «Рисунок» без сокращения и порядковый номер иллюстрации арабскими цифрами, например, «Рисунок 5» при сквозной нумерации или «Рисунок 4.2» при нумерации иллюстраций по разделам. Подпись иллюстраций, расположенных в приложениях, должна содержать слово «Рисунок», обозначение приложения и порядковый номер иллюстрации в приложении, например «Рисунок А.3». Иллюстрациям можно давать наименования, которые записываются после номера рисунка через знак тире с прописной буквы, точки после номера рисунка и после наименования не ставят, например: «Рисунок 3.2 – Блок-схема алгоритма».

Формулы в пояснительной записке, если их более одной, должны быть пронумерованы (независимо от наличия на них ссылок). Нумерация формул осуществляется в пределах раздела, номер формулы состоит из номера раздела и ее порядкового номера в разделе, разделенных точкой. Номер ставится в строках на уровне формул с правой стороны листа. Ссылки на формулы в тексте даются путем указания в круглых скобках номера формулы, например: «в соответст-

вии с формулой (3.1)» или «в уравнении (3.2)» и т. д. Значения символов и числовых коэффициентов, входящих в формулу, должны быть приведены с новой строки в той последовательности, в какой символы встречаются в формуле. Первая строка расшифровки начинается словом «где», без двоеточия после него.

Таблицы наглядно демонстрируют цифровые данные, используемые в работе. Оформление таблиц должно осуществляться в соответствии с требованиями ГОСТов. Все таблицы в тексте должны быть пронумерованы арабскими цифрами и текстовыми заголовками, причем слово «Таблица» не сокращают. Номер таблицы и заголовок разделяется знаком тире и располагается над таблицей. Слово «Таблица» пишется с левой границы поля, занимаемого таблицей. После слова «Таблица», порядкового номера и заголовка точки не ставятся. Заголовок пишется с прописной буквы. Строки с заголовком не должны выходить за пределы границ поля, занимаемого таблицей. Текст пояснительной записки и заголовок таблицы должны быть разделены пробельной строкой. Заголовок и материал таблицы пробельной строкой не разделяется. Таблица состоит из головки (шапки) и основной части. Если таблица не размещается на одной странице, то она продолжается на следующей странице, а в первой части таблицы нижняя горизонтальная линия, ограничивающая таблицу, не проводится. Продолжение таблицы оформляется начиная с повторения головки, над левым углом которой пишутся слова «Продолжение таблицы (номер)». Во второй части таблицы головку можно заменять соответствующими номерами граф. В этом случае в первой строке первой части таблицы необходимо осуществить нумерацию граф.

При оформлении приложений необходимо учитывать, что каждое приложение должно начинаться с новой страницы. Приложения обозначаются заглавными буквами русского алфавита, начиная с буквы «А», за исключением букв Ё, З, Й, О, Ч, Ь, Ъ, Ы. После слова ПРИЛОЖЕНИЕ следует буква, обозначающая его последовательность. Наверху посередине страницы пишется слово ПРИЛОЖЕНИЕ прописными буквами и его буквенное обозначение. Ниже в круглых скобках строчными буквами пишется слово «обязательное», «рекомендуемое» или «спра-

вочное». Приложение должно иметь заголовок, который записывается симметрично тексту с прописной буквы.

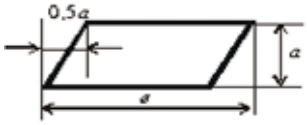
Слово СОДЕРЖАНИЕ записывается в виде заголовка прописными буквами и располагается симметрично тексту. Между словом СОДЕРЖАНИЕ и перечнем разделов оставляют одну пробельную строку. В содержании заголовки выравниваются по вертикалям разделов и подразделов. Вертикаль подразделов должна быть смещена относительно вертикали разделов на два знака. Если пункты имеют самостоятельный заголовок, то их вертикаль должна быть смещена относительно вертикали подразделов на пять знаков. Все заголовки в содержании должны начинаться с прописной буквы. Последнее слово каждого заголовка соединяется точками с соответствующим ему номером страницы в правом столбце содержания.

Общий объем пояснительной записки – 25–35 листов без учета приложений. Поля: слева – 30 мм, справа – 15 мм, сверху – 20 мм, снизу – 27 мм. Используемый шрифт должен быть аналогичен шрифту Times New Roman, размер 14 пт. Интервал между строк – полуторный.

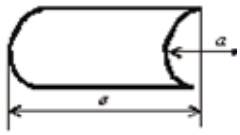
3 РЕКОМЕНДАЦИИ ПО ПОСТРОЕНИЮ БЛОК-СХЕМ АЛГОРИТМОВ

Основным требованием к блок-схемам алгоритмов является то, что они должны быть однозначно читаемы без каких-либо дополнительных пояснений со стороны разработчика. Это достигается выполнением блок-схем алгоритмов в соответствии с требованиями ГОСТа, которые устанавливают перечень допустимых условных графических обозначений. Правила построения блок-схем алгоритмов приводятся в ГОСТ 19-002-80 и ГОСТ 19-003-80. Согласно этим ГОСТам все размеры фигур связаны с двумя величинами: a и b, где a – величина, кратная 5, а b вычисляется по формуле b = 1,5a, допускается b = 2a. В январе 1992 г. введен новый ГОСТ 19-701-90. Он описывает, как и где следует использовать фигуры. Схемы алгоритмов состоят из имеющих заданное значение сим-

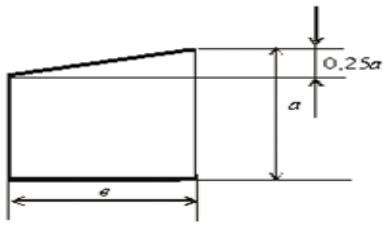
волов, краткого пояснительного текста и соединяющих линий. Описание символов:



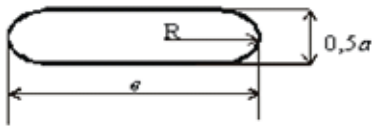
Символ отображает данные, носитель которых не определен



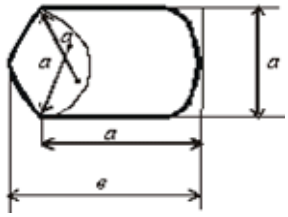
Символ отображает хранимые данные в виде, пригодном для обработки, носитель данных не определен



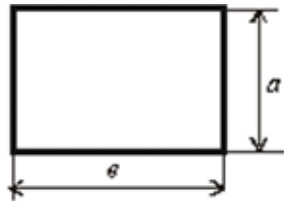
Ручной ввод. Символ отображает данные, вводимые вручную во время обработки с устройств любого типа (клавиатура, переключатели, кнопки, световое перо



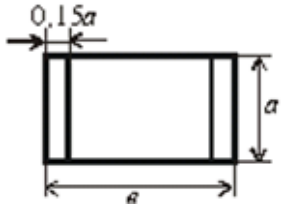
Символ отображает выход во внешнюю среду и вход из внешней среды (начало или конец схемы программы)



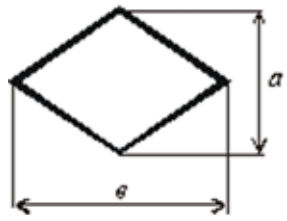
Символ отображает данные, представляемые в человекочитаемой форме на носителе в виде отображающего устройства (экран для визуального наблюдения)



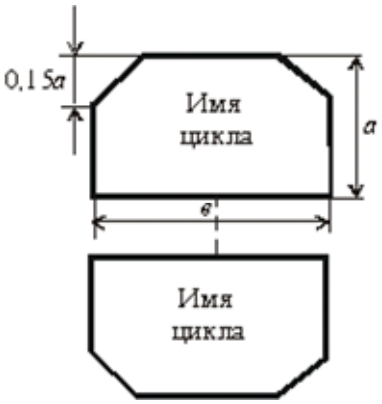
Символ отображает функцию обработки данных любого вида (выполнение определенной операции или группы операций, приводящее к изменению значения, формы или размещения информации)



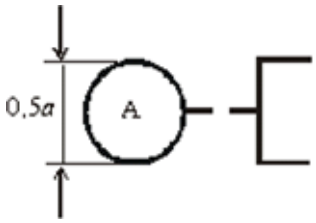
Символ отображает предопределенный процесс, состоящий из одной или нескольких операций или шагов программы, которые определены в другом месте (в подпрограмме, модуле)



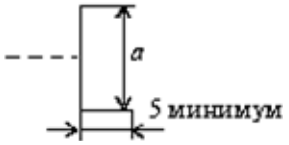
Символ отображает решение или функцию переключательного типа, имеющую один вход и ряд альтернативных выходов, один и только один из которых может быть активизирован после вычисления условий, определенных внутри этого символа. Соответствующие результаты вычисления могут быть записаны по соседству с линиями, отображающими эти пути



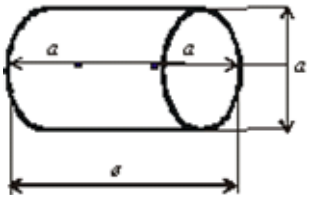
Символ, состоящий из двух частей, отображает начало и конец цикла. Обе части символа имеют один и тот же идентификатор. Условия для инициализации, приращения, завершения и т.д. помещаются внутри символа, в начале или в конце в зависимости от расположения операции, проверяющей условие



Символ отображает вход в часть схемы и выход из другой части схемы и используется для обрыва линии и продолжения ее в другом месте. Соответствующие символы-соединители должны содержать одно и то же уникальное обозначение



Символ используют для добавления описательных комментариев или пояснительных записей в целях объяснения или примечаний. Пунктирные линии в символе комментария связаны с соответствующим символом или могут обводить группу символов. Текст комментариев или примечаний должен быть помещен около ограничивающей фигуры



Символ отображает данные, хранящиеся в запоминающем устройстве с прямым доступом (магнитный диск, магнитный барабан, гибкий магнитный диск).

4 РЕКОМЕНДАЦИИ ПО ПОСТРОЕНИЮ ДИАГРАММЫ КЛАССОВ

Центральное место в объектно-ориентированном проектировании занимает разработка логической модели системы в виде диаграммы классов. Диаграмма классов (class diagram) служит для представления статической структуры модели в терминологии классов объектно-ориентированного программирования. Диаграмма классов может отражать различные взаимосвязи между отдельными сущностями предметной области, такими как объекты и подсистемы, а также описывает их внутреннюю структуру и типы отношений. От умения

правильно выбрать классы и установить между ними взаимосвязи часто зависит не только успех процесса проектирования, но и производительность выполнения программы. На диаграмме не указывается информация о временных аспектах функционирования системы. С этой точки зрения диаграмма классов является дальнейшим развитием концептуальной модели проектируемой системы.

Диаграмма классов представляет собой некоторый граф, вершинами которого являются элементы типа «классификатор», которые связаны различными типами структурных отношений. Такая диаграмма является статической структурной моделью проектируемой системы. Диаграмма классов состоит из множества элементов, которые в совокупности отражают декларативные знания о предметной области. Эти знания интерпретируются в базовых понятиях языка UML(унифицированный язык моделирования), таких, как классы, интерфейсы и отношения между ними и их составляющими компонентами. Модель системы должна быть согласована с внутренней структурой классов, которая описывается на языке UML.

Класс (class) в языке UML служит для обозначения множества объектов, которые обладают одинаковой структурой, поведением и отношениями с объектами из других классов. Графически класс изображается в виде прямоугольника, который дополнительно может быть разделен горизонтальными линиями на разделы или секции (рисунок 4.1). В этих разделах указываются имя класса, атрибуты (переменные) и операции (методы).

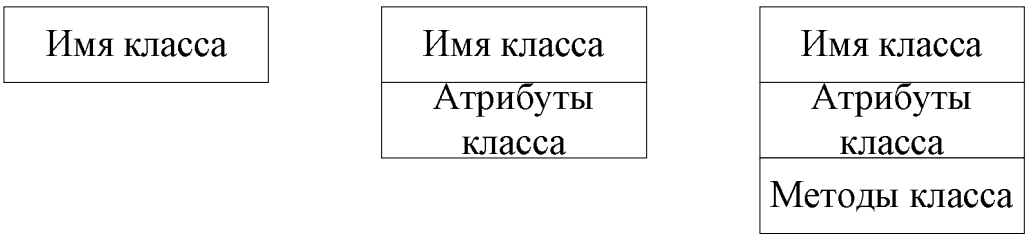


Рисунок 4.1 – Графическое изображение класса на диаграмме классов

На начальных этапах разработки диаграммы отдельные классы могут обозначаться простым прямоугольником с указанием только имени соответствующего класса. По мере разработки отдельных компонентов диаграммы описания классов дополняются атрибутами и операциями. Секция методов класса может быть пустой, но в обозначении класса она выделяется горизонтальной линией. Окончательный вариант диаграммы содержит наиболее полное описание классов, которые состоят из трех разделов или секций.

Имя класса должно быть уникальным, оно записывается по центру секции имени полужирным шрифтом и начинается с заглавной буквы. Имя абстрактного класса пишется курсивом.

Синтаксис описания атрибутов на языке UML имеет следующий вид:

модификатор_доступа имя: тип=значение_по_умолчанию

Здесь использованы следующие обозначения:

модификатор_доступа принимает следующие значения:

+ public

– private

protected

Синтаксические конструкции языка UML, предназначенные для описания методов, выглядят так:

модификатор_доступа имя(список параметров):
тип_возвращаемого_значения(строка свойств)

где

список параметров содержит параметры, разделенные запятой. Синтаксическая конструкция для описания параметров выглядит следующим образом:

направление имя: тип=значение_по_умочанию

Здесь элемент *направление* используется для индикации ввода (in), вывода (out), ввода-вывода (inout) параметра. Например:

+setTime(in hr: integer, in min: integer, in sec: integer)

Отношения между классами

На диаграмме классов указываются различные отношения между классами. Базовыми отношениями или связями в языке UML являются:

– отношение зависимости (dependency relationship);

– отношение ассоциации (association relationship);

– отношение обобщения (generalization relationship);

– отношение реализации (realization relationship).

Каждое из этих отношений имеет собственное графическое представление на диаграмме. Рассмотрим некоторые из них.

Отношение зависимости

Отношение зависимости используется в такой ситуации, когда некоторое изменение одного элемента модели может потребовать изменения другого элемента модели. Отношение зависимости графически изображается пунктирной линией между соответствующими элементами со стрелкой на одном из ее концов. На диаграмме классов данное отношение связывает отдельные классы между собой, при этом стрелка направлена от класса-клиента зависимости к независимому классу или классу-источнику (рисунок 4.2).

Рисунок 4.2 – Графическое изображение отношения зависимости

На рисунке 4.2 изображены два класса: Класс_А и Класс_Б, при этом Класс_Б является источником некоторой зависимости, а Класс_А – клиентом зависимости.

Отношение обобщения

Применительно к диаграмме классов данное отношение описывает иерархическое строение классов и наследование их свойств и поведения. При этом предполагается, что класс-потомок обладает всеми свойствами и поведе-

нием класса-предка, а также имеет свои собственные свойства и поведение, которые отсутствуют у класса-предка. На диаграмме отношение обобщения обозначается сплошной линией с треугольной стрелкой на одном из концов (рисунок 4.3). Стрелка указывает на класс-предок, а ее отсутствие – на класс-потомок.

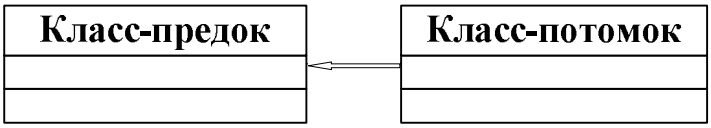


Рисунок 4.3 – Графическое изображение отношения обобщения

Интерфейсы

Интерфейсы являются элементами диаграммы вариантов использования, однако при построении диаграммы классов отдельные интерфейсы могут уточняться и в этом случае для их изображения используется специальный графический символ – прямоугольник класса с ключевым словом «interface» (рисунок 4.4). При этом секции атрибутов отсутствует, а указывается только секция операций.

Рисунок 4.4 – Графическое изображение интерфейса на диаграмме классов

Шаблоны или параметризованные классы

Шаблон (template) предназначен для обозначения такого класса, который имеет один (или более) нефиксированный параметр. Он определяет целое множество классов, каждый из которых может быть получен связыванием этих параметров с действительными значениями. Графически шаблон изображается прямоугольником, к верхнему правому углу которого присоединен маленький прямоугольник из пунктирных линий, большой прямоугольник может быть разделен на секции аналогично обозначению для класса. В верхнем прямоугольнике указывается список формальных параметров для тех классов, кото-

рые могут быть получены на основе данного шаблона. В верхней секции шаблона записывается его имя по правилам записи имен для классов (рисунок 4. 5).

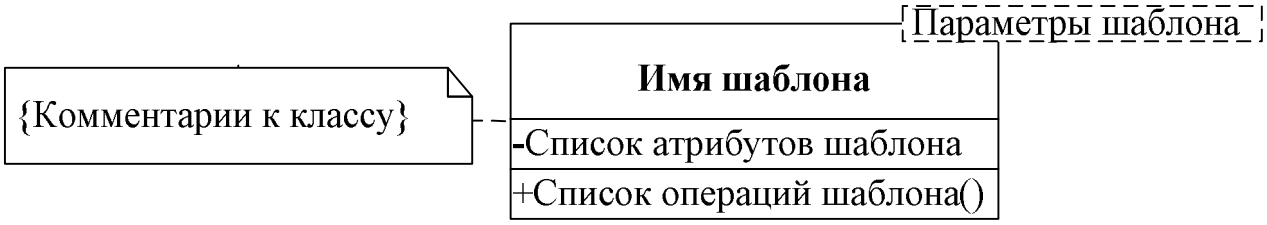


Рисунок 4.5 – Графическое изображение шаблона на диаграмме классов и комментарии к классу

5 ВАРИАНТЫ ЗАДАНИЙ НА КУРСОВУЮ РАБОТУ

Приведенные варианты заданий являются базовыми и по согласованию с преподавателем, осуществляющим руководство выполнением курсовой работы, могут быть расширены и дополнены. Номер варианта задания определяется индивидуально преподавателем, осуществляющим руководство выполнением курсовой работы.

1. Создать программу-картотеку видеофильмов. Программа должна иметь удобный пользовательский интерфейс с необходимыми пунктами меню. Работа с информацией должна производиться в окнах. Информация должна храниться в файле и включать: название фильма, описание фильма (жанр: боевик, триллер, комедия и т. д.), характеристику фильма (по 10-балльной системе), год выпуска, основных героев (дополнительную информацию можно добавить по желанию). Надо реализовать функции добавления новых поступлений, исключения старых фильмов, выдачи фильмов по героям, жанру, году выпуска и т. д. При реализации операций добавления, исключения необходимо предусмотреть операцию отмены последних действий. Необходимо реализовать оконный интерфейс при помощи ассемблерных вставок. Или реализовать ввод данных через собственные функции, реализованные на ассемблере.

2. Создать программу управления расписанием поездов. Программа должна иметь удобный пользовательский интерфейс с необходимыми пунктами меню. Работа с информацией должна производиться в окнах. В программе должно учитываться направление движения, дата, время отправления, стоимость билета. Программа должна по запросу выдавать информацию о наличии свободных мест, ближайших рейсах, о наличии рейсов в определенный день и время. При реализации операции редактирования, добавления, удаления информации необходимо предусмотреть операцию отмены последних действий. Необходимо реализовать оконный интерфейс при помощи ассемблерных вставок. Или реализовать ввод данных через собственные функции, реализованные на ассемблере.

3. Написать программу учета наличия транспортных средств (автобусы) в автопарке. Программа должна иметь удобный пользовательский интерфейс с необходимыми пунктами меню. Работа с информацией должна производиться в окнах. Информация организуется в виде структур и хранится в файле. Структура содержит следующие поля: марка автобуса, тип автобуса (мягкий/жесткий), состояние (на базе, в рейсе, в ремонте, списали), количество мест, государственный номер, водители. По запросу выдавать информацию о свободных автобусах, об автобусах в рейсе, ремонтируемых, списанных. Выдаваемая информация должна быть отсортирована по разным признакам. При реализации операции редактирования, добавления, удаления информации необходимо предусмотреть операцию отмены последних действий. Необходимо реализовать оконный интерфейс при помощи ассемблерных вставок. Или реализовать ввод данных через собственные функции, реализованные на ассемблере.

4. Написать программу учета наличия транспортных средств (грузовые машины) в автопарке. Программа должна иметь удобный пользовательский интерфейс с необходимыми пунктами меню. Работа с информацией должна производиться в окнах. Информация организуется в виде структур и хранится в файле. Структура содержит следующие поля: тип машины, текущее состояние

(в ремонте, списали, свободна, занята), грузоподъемность. По запросу выдавать информацию о свободных машинах, занятых, ремонтируемых, списанных. Выдаваемая информация должна быть отсортирована по задаваемому признаку. При реализации операции редактирования, добавления, удаления информации необходимо предусмотреть операцию отмены последних действий. Необходимо реализовать оконный интерфейс при помощи ассемблерных вставок. Или реализовать ввод данных через собственные функции, реализованные на ассемблере.

5. Написать программу ведения и учета коммерческих точек. Программа должна иметь удобный пользовательский интерфейс с необходимыми пунктами меню. Работа с информацией должна производиться в окнах. Создать файл, в котором хранятся сведения обо всех точках: 1) ассортимент товаров, 2) наличие разрешения на торговлю: дата выдачи, на какой срок, разрешение санэпидемстанции и т.д. Предусмотреть возможность регистрации новых точек, удаление информации о несуществующих, редактирование существующей информации. Выдача информации должна производиться в отсортированном виде. Сортировку осуществлять по разным признакам (тип продаваемого товара, дата выдачи разрешения и т. д.). При реализации операции редактирования, добавления, удаления информации необходимо предусмотреть операцию отмены последних действий. Необходимо реализовать оконный интерфейс при помощи ассемблерных вставок. Или реализовать ввод данных через собственные функции, реализованные на ассемблере.

6. Реализовать игру типа PACMAN. Сценарий игры состоит в том, что герой должен бегать по лабиринту и собирать призы. Когда все призы собраны, в лабиринте открывается дверь, куда может выйти герой и попасть на следующий уровень. За героем охотятся несколько злодеев. Если при хождении по лабиринту герой встречается со злодеями, он погибает. Должно быть реализовано не менее 10 уровней, возможность сохранения на конкретном уровне, подсчет очков и времени, затрачиваемого на уровень. Должна быть реализована сохра-

няемая таблица рекордов с сортировкой по очкам и затраченному времени. Сценарий игры может быть улучшен по желанию студента.

7. Написать программу управления авиарейсами. Программа должна иметь удобный пользовательский интерфейс с необходимыми пунктами меню. Работа с информацией должна производиться в окнах. Информация должна храниться в файле. В файле содержится информация об авиарейсах: откуда - куда, по каким дням, количество и тип (класс – первый, второй, третий) мест, стоимость билетов. Реализовать функции выдачи информации о нужном рейсе, добавления новой информации, резервирования билетов (и других по желанию). При реализации операции редактирования, добавления, удаления информации необходимо предусмотреть операцию отмены последних действий. Необходимо реализовать оконный интерфейс при помощи ассемблерных вставок. Или реализовать ввод данных через собственные функции, реализованные на ассемблере.

8. Написать программу составления посланий. Программа должна иметь удобный пользовательский интерфейс с необходимыми пунктами меню. Работа с информацией должна производиться в окнах. Необходимо предусмотреть ввод двух файлов. В первом хранятся адреса и фамилии девушек, а во втором – тексты посланий. Сделать функцию, чтобы любой выбранной девушке можно было послать любое пожелание (скоординированный текст). Должна быть возможность добавления, удаления и просмотра первого и второго файла. Созданные послания также должны помещаться в файл. При реализации операции редактирования, добавления, удаления информации необходимо предусмотреть операцию отмены последних действий. Необходимо реализовать оконный интерфейс при помощи ассемблерных вставок. Или реализовать ввод данных через собственные функции, реализованные на ассемблере.

9. Написать программу управления гостиничными номерами. Программа должна иметь удобный пользовательский интерфейс с необходимыми пунктами меню. Работа с информацией должна производиться в окнах. В программе

должна быть предусмотрена возможность создания информации о гостиничных номерах. Информация должна сохраняться в файле, в котором содержится информация по гостинице: ФИО, количество номеров, жилплощадь, класс номеров, стоимость номеров (и другая по желанию). Написать функции: вывода на экран информации о постояльцах, добавления, удаления и резервирования номеров. Предусмотреть санкционированный доступ к информации по паролю. При реализации операции редактирования, добавления, удаления информации необходимо предусмотреть операцию отмены последних действий. Необходимо реализовать оконный интерфейс при помощи ассемблерных вставок. Или реализовать ввод данных через собственные функции, реализованные на ассемблере.

10. Написать программу управления информацией в автомобильном магазине. Программа должна иметь удобный пользовательский интерфейс с необходимыми пунктами меню. Работа с информацией должна производиться в окнах. В программе должна быть предусмотрена возможность создания информации в виде файла, где хранится список всех автомобилей и их характеристики. Также должен быть предусмотрен второй файл – список автостанций, где автомобили той или иной марки обслуживаются. По желанию можно узнать о наличии какой-то машины, если имеется, то вывести ее характеристику. А также узнать, где какой автомобиль обслуживается. Необходимо реализовать оконный интерфейс при помощи ассемблерных вставок. Или реализовать ввод данных через собственные функции, реализованные на ассемблере.

11. Написать графический редактор с поддержкой формата BMP, а также JPEG или GIF. В редакторе необходимо реализовать функции просмотра, создания, сохранения рисунка в указанном формате. Необходимо реализовать графический интерфейс с поддержкой «мыши». Из графических примитивов необходимо реализовать линию, прямоугольник (с закраской и без), круг (с закраской и без), дугу, треугольник (с закраской и без), точку (и другие по желанию). Необходимо предусмотреть операцию отмены последних действий (более од-

ного). Необходимо реализовать прорисовку графических примитивов при помощи ассемблерных вставок.

12. Написать игру. Сценарий игры должен быть схожим со сценарием игры Xonix. Необходимо закрашивать прямоугольные области экрана, перемещаясь по горизонтали и вертикали. Игрока преследуют противники, при столкновении с которыми игрок погибает. В закрашиваемой области, как в аквариуме, также находятся противники, они могут столкнуться с игроком при осуществлении операции закрашивания. Должно быть реализовано не менее 10 уровней, возможность сохранения на конкретном уровне, подсчет очков и времени, затрачиваемого на уровень. Должна быть реализована сохраняемая таблица рекордов с сортировкой по очкам и затраченному времени. Сценарий игры может быть улучшен по желанию студента.

13. Написать программу учета книг. Программа должна иметь удобный пользовательский интерфейс с необходимыми пунктами меню. Работа с информацией должна производиться в окнах. В программе должна быть предусмотрена возможность создания информации в виде файла, где будут храниться: ФИО студента, факультет, группа и название файла, в котором содержится название литературы, имеющейся у студента, и время сдачи. Реализовать: 1) ввод, вывод литературы и данных о студентах; 2) вывод данных о студентах, имеющих данную книгу; 3) вывод данных о студентах и литературу, у которой просрочено время сдачи. Ввод литературы обеспечить через код (для оптимизации). При реализации операции редактирования, добавления, удаления информации необходимо предусмотреть операцию отмены последних действий. Необходимо реализовать оконный интерфейс при помощи ассемблерных вставок. Или реализовать ввод данных через собственные функции, реализованные на ассемблере.

14. Написать программу учета радиодеталей. Программа должна иметь удобный пользовательский интерфейс с необходимыми пунктами меню. Работа с информацией должна производиться в окнах. В программе должна быть пре-

дусмотрена возможность создания информации в виде файлов. В головном файле находятся имена районов, в которых находятся магазины. Файл определенного магазина содержит список секций. Файл секций содержит список товара. Реализовать действия: добавление товаров в магазин, покупка товаров, просмотр по определенному признаку и т.д. Необходимо реализовать оконный интерфейс при помощи ассемблерных вставок. Или реализовать ввод данных через собственные функции, реализованные на ассемблере.

15. Написать программу ведения и учета наличия мест в детских садах конкретного района. Программа должна иметь удобный пользовательский интерфейс с необходимыми пунктами меню. Работа с информацией должна производиться в окнах. В программе должна быть предусмотрена возможность создания информации в виде файлов. Имеется файл, в записях которого указываются номер детсада и имя файла с информацией о состоянии этого детсада. Состояние детсада характеризуется количеством групп, мест в группе, городской или ведомственной принадлежностью. По запросу выдать: 1) номера детсадов данного района; 2) информацию о конкретном детсаде: 3) наличие определенного типа мест (ясли, младшая, средняя и старшая группы) в детсадах; 4) о состоянии ближайшего детсада (свободные места). Необходимо реализовать оконный интерфейс при помощи ассемблерных вставок. Или реализовать ввод данных через собственные функции, реализованные на ассемблере.

16. Написать программу-картотеку аудиозаписей. Программа должна иметь удобный пользовательский интерфейс с необходимыми пунктами меню. Работа с информацией должна производиться в окнах. Информация должна храниться в файле и включать: название аудиозаписи, описание аудиозаписи, характеристика аудиозаписи (по 10-балльной системе), год выпуска, исполнитель (дополнительную информацию можно добавить по желанию). Надо реализовать функции добавления новых поступлений, исключения старых аудиозаписей, выдачи аудиозаписи по исполнителям, жанру, году выпуска и т. д. При реализации операций добавления, исключения необходимо предусмотреть опе-

рацию отмены последних действий. Необходимо реализовать оконный интерфейс при помощи ассемблерных вставок. Или реализовать ввод данных через собственные функции, реализованные на ассемблере.

17. Написать программу реализации банковских услуг. Программа должна иметь удобный пользовательский интерфейс с необходимыми пунктами меню. Работа с информацией должна производиться в окнах. В программе должна быть предусмотрена возможность создания информации в виде файлов. Программа позволяет организовать открытие нового счета, аннулирование счета, перевод денег с одного счета на другой, снятие денег со счета, депозиты и займ под проценты. Все данные содержатся в файлах. Банковский счет связывается с именем вкладчика. Необходимо реализовать оконный интерфейс при помощи ассемблерных вставок. Или реализовать ввод данных через собственные функции, реализованные на ассемблере.

18. Написать программу, управляющую работой библиотеки. Программа должна иметь удобный пользовательский интерфейс с необходимыми пунктами меню. Работа с информацией должна производиться в окнах. Реализовать функции: добавление, удаление книг из отделов, выдача книг на абонемент. Информация должна храниться в различных файлах. Реализовать классы: книга, отдел, библиотека. Необходимо реализовать оконный интерфейс при помощи ассемблерных вставок. Или реализовать ввод данных через собственные функции, реализованные на ассемблере.

19. Реализовать информационную систему для компаний по продаже недвижимости. Программа должна иметь удобный пользовательский интерфейс с необходимыми пунктами меню. Работа с информацией должна производиться в окнах. Информация должна храниться в различных файлах. Необходимо хранить следующие сведения: данные о квартирах, частном секторе, нежилых помещениях, коммерческой недвижимости, а также о клиентах. Реализовать функции: добавление, удаление, редактирование данных, поиск информации по заданным признакам. При реализации функций добавление, удаление, редак-

тирование данных необходимо предусмотреть операцию отмены последних действий (более одного). Необходимо реализовать оконный интерфейс при помощи ассемблерных вставок. Или реализовать ввод данных через собственные функции, реализованные на ассемблере.

20. Реализовать информационную систему по начислению заработной платы работникам предприятия. Программа должна иметь удобный пользовательский интерфейс с необходимыми пунктами меню. Работа с информацией должна производиться в окнах. Информация должна храниться в различных файлах. Реализовать функции создание отделов, описание и ввод должностей, ввод фамилий, окладов, надбавок, премий. Реализовать бизнес-логику по обработке данных в случаях ухода в отпуск, приема и увольнения работников. Разработать и использовать в программе классы контейнеров и итераторов. Необходимо реализовать оконный интерфейс при помощи ассемблерных вставок. Или реализовать ввод данных через собственные функции, реализованные на ассемблере.

21. Написать программу для обслуживания клиентов магазина. Программа должна иметь удобный пользовательский интерфейс с необходимыми пунктами меню. Работа с информацией должна производиться в окнах. Информация должна храниться в различных файлах, при этом каждая группа товаров должна иметь отдельный файл. Каждый товар имеет характеристики (группа, тип, индивидуальные особенности, страна происхождения и т. д.) и штрих-код. При обслуживании клиента необходимо подготовить электронный чек, в котором должно быть указано название товара, его цена, количество, общая сумма покупки, дата и время покупки. Чеки должны сохраняться в файлы. Все покупки, совершенные клиентами, должны записываться в файл. Разработать и использовать в программе классы контейнеров и итераторов. Необходимо реализовать оконный интерфейс при помощи ассемблерных вставок. Или реализовать ввод данных через собственные функции, реализованные на ассемблере.

22. Разработать информационную систему по продаже компьютерных комплектующих. Программа должна иметь удобный пользовательский интерфейс с необходимыми пунктами меню. Работа с информацией должна производиться в окнах. Информация должна храниться в различных файлах. Реализовать функции: добавление, редактирование, удаление данных по комплектующим, поиск данных по различным признакам. Разработать и использовать в программе классы контейнеров и итераторов. Необходимо реализовать оконный интерфейс при помощи ассемблерных вставок. Или реализовать ввод данных через собственные функции, реализованные на ассемблере.

23. Разработать информационную систему кинотеатра. Программа должна иметь удобный пользовательский интерфейс с необходимыми пунктами меню. Работа с информацией должна производиться в окнах. Информация должна храниться в различных файлах. При описании фильма должна вноситься информация о жанре фильма, об исполнителях главных ролей, режиссерах. Необходимо вести статистику о просмотренных фильмах. По запросу необходимо выдавать по заданным признакам статистическую информацию о фильмах на экран. Разработать и использовать в программе классы контейнеров и итераторов. Необходимо реализовать оконный интерфейс при помощи ассемблерных вставок. Или реализовать ввод данных через собственные функции, реализованные на ассемблере.

24. Разработать визуальный редактор HTML-кода. Предусмотреть многооконный режим редактирования файлов. Реализовать в редакторе следующие функции: возможность формирования текста на базе стандартных стилей, создание произвольных пользовательских стилей, возможность логического форматирования текста (подчеркивание, курсив, жирный шрифт), возможность вставки в документ картинок, гиперссылок, ненумерованных списков.

25. Разработать программу адресной книги с поддержкой фотографий абонентов (формат JPEG и GIF). Программа должна иметь удобный пользовательский интерфейс с необходимыми пунктами меню. Работа с информацией

должна производиться в окнах. Информация должна храниться в различных файлах. Кроме адреса и телефона предусмотреть хранение дополнительной информации об абоненте. Реализовать функции: добавление, удаление, редактирование, сортировка записей по разным признакам, разбиение записей на группы: друзья, коллеги и т. д. Предусмотреть возможность поиска человека по имени, телефону и т. д. Необходимо реализовать оконный интерфейс при помощи ассемблерных вставок. Или реализовать ввод данных через собственные функции, реализованные на ассемблере.

ЛИТЕРАТУРА

1 Дейтел, Х. М. Как программировать на C++ / Х. М. Дейтел, П. Д. Дейтел; пер. с англ. – М. : Бином, 2007.

2 Страуструп, Б. Язык программирования C++ / Б. Страуструп; специальное издание; пер. с англ. – СПб. : BHV, 2008.

3 Скляров, В. А. Язык C++ и объектно-ориентированное программирование: справ. пособие / В. А. Скляров. – Минск : Выш. шк., 1997.

4 Элджер, Дж. C++: библиотека программиста / Дж. Элджер. – СПб. : Питер, 2001.

5 Шилд, Г. Программирование на Borland C++ для профессионалов / Г. Шилд. – Минск : ООО «Попури», 1998.

6 Буч, Г. Язык UML. Руководство пользователя / Г. Буч, Д. Рамбо, А. Джекобсон; пер. с англ. – СПб. : ДМК, 2004.

7 Доманов, А. Т. Предварительный стандарт предприятия. Дипломные проекты (работы) : общие требования / А. Т. Доманов, Н. И. Сороко. – Минск : БГУИР, 2009.

Св. план 2009, поз. 62

Учебное издание

КОНСТРУИРОВАНИЕ ПРОГРАММ И ЯЗЫКИ ПРОГРАММИРОВАНИЯ

МЕТОДИЧЕСКИЕ УКАЗАНИЯ
ПО КУРСОВОМУ ПРОЕКТИРОВАНИЮ
для студентов специальности 1-40 02 01
«Вычислительные машины, системы и сети»
всех форм обучения

С о с т а в и т е л и:

Бушкевич Алексей Владимирович
Ковальчук Анна Михайловна
Лукьянова Ирина Викторовна

Редактор Т. П. Андрейченко

Корректор Е. Н. Батурчик

Подписано в печать 26.01.2010.	Формат 60x84 1/16.	Бумага офсетная.
Гарнитура «Таймс».	Отпечатано на ризографе.	Усл. печ. л. 1,97.
Уч.-изд. л. 1,5.	Тираж 150 экз.	Заказ 118.

Издатель и полиграфическое исполнение: Учреждение образования
«Белорусский государственный университет информатики и радиоэлектроники»
ЛИ №02330/0494371 от 16.03.2009. ЛП №02330/0494175 от 03.04.2009.
220013, Минск, П. Бровки, 6