МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования
«Курганский государственный университет»

Кафедра программного обеспечения автоматизированных систем

ОСНОВЫ ПРОГРАММИРОВАНИЯ

Методические указания
к выполнению лабораторных и практических работ
для студентов направления подготовки 09.03.04
«Программная инженерия»,
10.05.03 «Информационная безопасность автоматизированных систем»
Часть 2

Курган 2016

Кафедра: «Программное обеспечение автоматизированных систем»

Направление подготовки: 09.03.04 «Программная инженерия»,
10.05.03. «Информационная безопасность автоматизированных систем».

Дисциплина: «Основы программирования».

Составил: канд. техн. наук, доцент А.М. Семахин.

Утверждены на заседании кафедры «30» сентября 2015 г.

Рекомендованы методическим советом университета «17» декабря 2015 г.

ВВЕДЕНИЕ

Визуальное программирование использует технологию визуального проектирования и событийного программирования. Эта технология позволяет создавать диалоговые окна (визуальное проектирование) и функции обработки событий (событийное программирование).

В части 2 методических указаний приводятся теоретические сведения по визуальному программированию, основам структур и поиску данных, сверхдлинной целочисленной арифметике и стандартной библиотеке шаблонов. Для закрепления теоретических знаний и приобретения практических навыков визуального и обобщённого программирования приводятся варианты лабораторных и практических работ.

Лабораторный практикум (34 часов).

Практические работы (16 часов).

Методические указания разработаны в соответствии с требованиями государственного образовательного стандарта по подготовке бакалавров по направлениям 09.03.04 «Программная инженерия» и 10.05.03 «Информационная безопасность автоматизированных систем».

1 Разработка визуальных приложений в среде программирования Microsoft Visual C++

1.1 Базовые компоненты

Среда программирования Microsoft Visual C++ позволяет разрабатывать приложения Windows Forms (.NET). Платформа .NET Framework включает два компонента:

- общеязыковую исполняющую среду (Common Language Runtime, CLR);
- библиотеку классов .NET Framework.

Библиотека классов .NET Framework обеспечивает функциональную поддержку программного кода, выполняемого независимо от применяемого языка программирования [1, с. 28].

Библиотека классов .NET Framework включает:

- ADO.NET (ActiveX Data Object .NET) – технологию, предоставляющую доступ к данным для приложений Microsoft .NET;
- ASP.NET (Active Server Page .NET) – технологию создания web-приложений и веб-сервисов Microsoft .NET;
- Windows Forms – библиотеку разработки компонентов графического интерфейса пользователя;
- Windows Presentation Foundation (WPF) – графическую подсистему в составе .NET Framework, использующую язык XAML (eXtensible Application Markup Language – расширяемый язык разметки для приложений) [2, с. 9].

Компонент (component) – объект, предназначенный для решения задачи.

Компоненты обеспечивают ввод данных, отображение результатов, доступ к базам данных, решение других задач [3, с. 61].

К базовым (основным) компонентам относят:

- Label – поле отображения информации;
- TextBox – поле ввода-редактирования текста (данных);
- Button – командную кнопку;
- CheckBox – флажок;
- RadioButton – радиокнопку;
- ListBox – список выбора;
- ComboBox – поле редактирования со списком выбора [4, с. 5].

Компоненты располагаются в палитре компонентов.

1.1.1 Компоненты ввода-вывода

Для выполнения ввода-вывода данных применяются компоненты **Label**, **TextBox, Button, ComboBox, NumericUpDown**.

Компонент **Label** предназначен для отображения текстовой информации. Задать текст можно во время разработки формы и во время работы программы, присвоив значение свойству text [4, с. 227].

Компонент **TextBox** предназначен для ввода данных с клавиатуры. В поле редактирования вводится одна или несколько строк текста [4, с. 238].

Компонент **Button** предназначен для создания командной кнопки [4, с. 219].

Компонент **ComboBox** предназначен для ввода данных посредством набора на клавиатуре или выбором из списка. Компонент представляет комбинацию поля редактирования и списка [4, с. 221].

Компонент **NumericUpDown** предназначен для ввода числовых данных. Данные можно ввести в поле редактирования путем набора на клавиатуре или изменить введенные данные при помощи командных кнопок <Увеличить> и <Уменьшить>, которые находятся справа от поля редактирования [4, с. 230].

1.1.2 Лабораторная работа №1 «Визуальное программирование линейных процессов в Microsoft Visual C++»

Разработайте программу на языке C++, используя технологию визуального проектирования и событийного программирования.

Вариант 1. Пересчет веса из фунтов в килограммы (1 фунт равен 405,9 грамма).

Вариант 2. Пересчет расстояния из верст в километры (1 верста равна 1066,8 м).

Вариант 3. Расчет дохода по вкладу. Процентная ставка (% годовых) и время хранения (дней) задаются во время работы программы.

Вариант 4. Пересчет величины временного интервала, заданного в минутах, в величину, выраженную в часах и минутах.

Вариант 5. Преобразование введенного с клавиатуры дробного числа в денежный формат. Например, число 12.5 преобразуется к виду 12 руб. 50 коп.

Вариант 6. Расчет сопротивления электрической цепи, состоящей из двух последовательно соединенных сопротивлений.

Вариант 7. Расчет силы тока в электрической цепи.

Вариант 8. Расчет сопротивления электрической цепи, состоящей из двух параллельно соединенных сопротивлений.

Вариант 9. Расчет площади параллелограмма.

Вариант 10. Расчет объема параллелепипеда.

Методические указания

1 Создайте проект «Windows Forms Application Visual C++».
2 Разработайте диаграмму классов.
3 Формализуйте алгоритм решения задачи на ПЭВМ.
4 Используйте компоненты ввода-вывода.
5 Выведите на экран видеомонитора результаты работы программы.
6 Оформите отчет по лабораторной работе.

1.1.3 Компоненты выбора

Для выбора элементов (опций) применяются компоненты **RadioButton, CheckBox, CheckedListBox.**

Компонент **RadioButton** предназначен для выбора элемента из нескольких альтернатив. Компонент RadioButton представляет переключатель, состояние которого зависит от состояния других переключателей. Один из переключателей группы может находиться в выбранном состоянии. Состояние компонентов, принадлежащих одной группе, не зависит от состояния компонентов, принадлежащих другой группе [4, с. 234].

Компонент **GroupBox** предназначен для объединения в группы компонентов **RadioButton** по функциональному признаку. Компонент представляет контейнер для других компонентов [4, с. 225].

Компонент **CheckBox** предназначен для выбора нескольких опций из ряда возможных. Компонент представляет флажок, который находится в одном из двух состояний: выбранном или невыбранном. Поясняющий текст приводится рядом с флажком [3, с. 341].

Компонент **CheckedListBox** предназначен для выбора опций из предлагаемого перечня. Компонент представляет список, перед каждым элементом которого находится переключатель **CheckBox** [4, с. 224].

1.1.4 Лабораторная работа №2
«Визуальное программирование процессов ветвления в Microsoft Visual C++»

Разработайте программу расчета $y_1 = f_1(x)$, $y_2 = f_2(x)$, используя визуальное программирование.

Вариант 1. $y_1 = \cos x + \sin x + \cos^3 x + \sin^3 x$

$$y_2 = \frac{1 - tg\alpha}{1 + tg\alpha}$$

Вариант 2. $y_1 = \cos^2(\frac{3}{8} * \pi - \frac{x}{4}) - \cos^2(\frac{11}{8} * \pi + \frac{x}{4})$

$$y_2 = \frac{\sqrt{2}}{2} * \sin\frac{x}{2}$$

Вариант 3. $y_1 = (\cos x_1 - \cos x_2)^2 - (\sin x_1 - \sin x_2)^2$

$$y_2 = -4 * \sin^2 * \frac{x_1 - x_2}{2} * \cos(x_1 + x_2)$$

Вариант 4. $y_1 = \frac{\sin 4 * x}{1 + \cos 4 * x} * \frac{\cos 2 * x}{1 + \cos 2 * x}$

$$y_2 = tg3 * x$$

Вариант 5. $y_1 = \frac{\sin x_1 + \cos(2 * x_2 - x_1)}{\cos x_1 - \sin(2 * x_2 - x_1)}$

$$y_2 = \sqrt{\frac{x+3}{x-3}}$$

Вариант 6. $y_1 = \frac{\sqrt{2 * x + 2 * \sqrt{x^2 - 4}}}{\sqrt{x^2 - 4 + x + 2}}$

$$y_2 = \frac{1}{\sqrt{x+2}}$$

Вариант 7. $y_1 = \frac{\sqrt{(3 * x + 2)^2 - 24 * x}}{3 * \sqrt{x} - \frac{2}{\sqrt{x}}}$

$$y_2 = -\sqrt{x}$$

Вариант 8. $y_1 = \frac{\sin 4 * x}{1 + \cos 4 * x} * \frac{\cos 2 * x}{1 + \cos 2 * x}$

$$y_2 = tg2 * x + \sec 2 * x$$

Вариант 9. $y_1 = (\frac{x+2}{\sqrt{2 * x}} - -\frac{x}{\sqrt{2 * x} + 2} + \frac{2}{x - \sqrt{2 * x}}) * \frac{\sqrt{x} - \sqrt{2}}{x + 2}$

$$y_2 = \frac{1}{\sqrt{x} + \sqrt{2}}$$

Вариант 10. $y_1 = (\frac{1 + x + x^2}{2 * x + x^2} + 2 - \frac{1 - x + x^2}{2 * x - x^2})^{-1} * (5 - 2 * x^2)$

$$y_2 = \frac{4 - x^2}{2}$$

Методические указания

1 Создайте проект «Windows Forms Application Visual C++».
2 Разработайте диаграмму классов.
3 Формализуйте алгоритм решения задачи на ПЭВМ.
4 Используйте компоненты выбора.
5 Выведите на экран видеомонитора результаты работы программы.
6 Оформите отчет по лабораторной работе.

1.1.5 Компоненты вывода списка элементов

Для вывода списка элементов применяются компоненты **ListView**, **ListBox**.

Компонент **ListView** предназначен для наглядного представления списков [3, с. 103]. С его помощью в форме выводится список элементов с пиктограммами [5, с. 274].

Компонент **ListBox** представляет список, в котором можно выбрать нужный элемент [3, с. 98].

1.1.6 Лабораторная работа №3 «Визуальное программирование циклических процессов в Microsoft Visual C++»

Разработайте программу вывода табличных значений x и $y = f(x)$, используя визуальное программирование. Переменная x меняется от $x_н$ до $x_к$ с шагом h.

Вариант 1. $y = \frac{\sin 2 * x + \sin 5 * x - \sin 3 * x}{\cos x - \cos 3 * x + \cos 5 * x}$

Вариант 2. $y = \frac{\sin 4 * x}{1 + \cos 4 * x} * \frac{\cos 2 * x}{1 + \cos 2 * x}$

Вариант 3. $y = \frac{\sqrt{2 * x + 2 * \sqrt{x^2 - 4}}}{\sqrt{x^2 - 4 + x + 2}}$

Вариант 4. $y = (\frac{x + 2}{\sqrt{2 * x}} - - \frac{x}{\sqrt{2 * x} + 2} + \frac{2}{x - \sqrt{2 * x}}) * \frac{\sqrt{x} - \sqrt{2}}{x + 2}$

Вариант 5. $y = \dfrac{\sin 2 * x + \sin 5 * x - \sin 3 * x}{\cos x - \cos 3 * x + \cos 5 * x}$

Вариант 6. $y = \dfrac{(x_1 - 1) * \sqrt{x_1} - (x_2 - 1) * \sqrt{x_1}}{\sqrt{{x_1}^3 * x_2} + x_2 * x_1 + {x_1}^2 - x_1}$

Вариант 7. $y = \dfrac{\sin 4 * x}{1 + \cos 4 * x} * \dfrac{\cos 2 * x}{1 + \cos 2 * x}$

Вариант 8. $y = (\dfrac{1 + x + x^2}{2 * x + x^2} + 2 - \dfrac{1 - x + x^2}{2 * x - x^2})^{-1} * (5 - 2 * x^2)$

Вариант 9. $y = \cos^4 x + \sin^2 y + \dfrac{1}{4} * \sin^2 2 * x - 1$

Вариант 10. $y = 2 * \sin^2 * (3 * \pi - 2 * x) * \cos^2 (5 * \pi + 2 * x)$

Методические указания

1 Создайте проект «Windows Forms Application Visual C++».
2 Разработайте диаграмму классов.
3 Формализуйте алгоритм решения задачи на ПЭВМ.
4 Используйте компоненты вывода списка элементов.
5 Выведите на экран видеомонитора результаты работы программы.
6 Оформите отчет по лабораторной работе.

1.1.7 Практическая работа №1 «Расчет значений функции при изменении входной переменной с шагом в заданном диапазоне»

Разработайте визуальное приложение, рассчитывающее выходную переменную y функции для аргумента x, изменяющегося на интервале от $X_н$ до $X_к$ с шагом h, где a, b, c –действительные числа.

Вариант 1

$$y = \begin{cases} a * x^2 + b, при\ x - 1 < 0, \text{b - x} \neq 0 \\ \dfrac{\text{x - a}}{\text{x}}, при\ x - 1 > 0\ и\ b + x = 0 \\ \dfrac{x}{c}, в\ остальных\ случаях \end{cases}$$

Вариант 2

$$y=\begin{cases} -a*x^3-b, при\ x+c<0, \text{a}\neq 0 \\ \dfrac{\text{x - a}}{\text{x - c}}, при\ x+c>0\ и\ a=0 \\ \dfrac{x}{c}+\dfrac{c}{x}, в\ остальных\ случаях \end{cases}$$

Вариант 3

$$y=\begin{cases} -a*x^2+b, при\ x<0, \text{b}\neq 0 \\ \dfrac{\text{x}}{\text{x - c}}+5.5, при\ x>0\ и\ b=0 \\ \dfrac{x}{-c}, в\ остальных\ случаях \end{cases}$$

Вариант 4

$$y=\begin{cases} a*(x+c)^2-b, при\ x=0, \text{b}\neq 0 \\ \dfrac{\text{x - a}}{\text{- c}}, при\ x=0\ и\ \text{b}=0 \\ a+\dfrac{x}{c}, в\ остальных\ случаях \end{cases}$$

Вариант 5

$$y=\begin{cases} a*x^2-cx+b, при\ x+10<0, \text{b}\neq 0 \\ \dfrac{\text{x - a}}{\text{x - c}}, при\ x+10>0\ и\ b=0 \\ \dfrac{-x}{a-c}, в\ остальных\ случаях \end{cases}$$

Вариант 6

$$y=\begin{cases} a*x^3+b*x^2, при\ x<0, \text{b}\neq 0 \\ \dfrac{\text{x - a}}{\text{x - c}}, при\ x>0\ и\ b=0 \\ \dfrac{x+5}{c(x-10)}, в\ остальных\ случаях \end{cases}$$

Вариант 7

$$y=\begin{cases} a*(x+7)^2-b, при\ x<5, \mathrm{b}\neq 0 \\ \dfrac{\mathrm{x}-\mathrm{c}*\mathrm{d}}{\mathrm{a}*\mathrm{x}}, при\ x>5\ и\ b=0 \\ \dfrac{x}{c}, в\ остальных\ случаях \end{cases}$$

Вариант 8

$$y=\begin{cases} -\dfrac{2*x-c}{c*x-a}, при\ x<0, \mathrm{b}\neq 0 \\ \dfrac{\mathrm{x}-\mathrm{a}}{\mathrm{x}-\mathrm{c}}, при\ x>0\ и\ b=0 \\ -\dfrac{x}{c}+\dfrac{-b}{2*c}, в\ остальных\ случаях \end{cases}$$

Вариант 9

$$y=\begin{cases} a*x^3+b^2+c, при\ x<0, \mathrm{b}\neq 0 \\ \dfrac{\mathrm{x}-\mathrm{a}}{\mathrm{c}}, при\ x>0\ и\ c\neq 0 \\ \dfrac{15*x}{c}, в\ остальных\ случаях \end{cases}$$

Вариант 10

$$y=\begin{cases} a*x^2+2*b*x+5, при\ x<5, \mathrm{b}\neq 0 \\ \dfrac{3*\mathrm{x}-\mathrm{a}}{\mathrm{c}}, при\ x\geq 5\ и\ c\neq 0 \\ \dfrac{10*x}{c+6}, в\ остальных\ случаях \end{cases}$$

1.2 Работа с базами данных

1.2.1 Локальная и распределённая базы данных

Среда программирования Microsoft Visual C++ имеет набор компонентов для работы с базами данных Microsoft Access, Microsoft SQL Server и Oracle.

База данных (data base) – это файл или совокупность файлов определённой структуры, в которых находится информация [3, с. 197].

Базы данных подразделяются на локальные (Microsoft Access) и удалённые (Microsoft SQL Server Compact Edition). В удалённых файлы данных размещают на отдельном компьютере (сервере).

Работу с базой данных Microsoft SQL Server обеспечивают компоненты:

- oleDbConnection – соединение с базой данных (сервером);
- oleDbDataAdapter – взаимодействие с базой данных путем отправки SQL-команд серверу;
- dataSet – хранение информации, полученной из базы данных;
- dataGridView – отображение информации, полученной из базы данных, выполнение операций редактирования, добавления и удаления записей [4, с. 116].

1.2.2 Лабораторная работа №4 «Визуальное приложение с использованием базы данных»

Вариант 1. Институт селекции занимается сбором, выведением и продажей сортов семян. В ассортименте можно найти семена всех видов растений. Выведенные сорта заносятся в список для дальнейшего тестирования. Каждый сорт семян имеет свои характеристики, такие как урожайность, морозоустойчивость, адаптация к местным условиям, сроки созревания (раннеспелый, среднеспелый, поздний). Покупатель может выбрать сорт, отвечающий характеристикам. Компания занимается оптовыми и розничными продажами.

Вариант 2. Университет зачисляет абитуриентов после сдачи вступительных экзаменов. На бюджетную основу зачисляются абитуриенты, получившие высокий балл ЕГЭ и успешно прошедшие собеседование, абитуриенты, набравшие необходимый для бесплатного поступления балл на университетских экзаменах, и абитуриенты, имеющие направление от государственного предприятия. Все остальные могут поступить в университет на платной основе, набрав установленное университетом необходимое число баллов на вступительных экзаменах.

Вариант 3. Касса авиакомпании занимается продажей билетов на предстоящие рейсы. В билете указывается номер и название рейса, номер кассы и данные (дата и время вылета, прибытия, номер места и класс). Цена билета зависит от рейса, лайнера, класса и времени покупки билета.

Вариант 4. Предприятие по учету электроэнергии взымает плату с каждой квартиры в зависимости от количества потребленной энергии. Плата зависит от вида счетчика (однофазный, трехфазный), типа счетчика (возможность учета дневного и ночного тарифов) и вида квартиры (коммунальная, отдельная). Квартиросъемщик ежемесячно снимает показания счетчика и производит оплату за потребленную электроэнергию через сбербанк.

Вариант 5. Судоходная компания «Балтика» занимается перевозками грузов между континентами. В ее собственности несколько десятков судов различного класса и грузоподъемности. К услугам компании обращаются клиенты различных стран мира. На судне может находиться несколько партий грузов для различных грузополучателей из различных стран и городов. Одна партия груза может состоять из нескольких разновидностей грузов. У одной

партии груза может быть только один отправитель и только один получатель. Маршрут следования судна разрабатывается главным менеджером компании и проходит через несколько портов. В порту назначения производится частичная погрузка и выгрузка грузов, и судно следует дальше.

Вариант 6. Учреждение юстиции регистрирует права юридических и физических лиц на недвижимое имущество (здания, квартиры, земельные участки). Разработайте программу регистрации прав граждан на квартиры. В здании несколько квартир. В одной квартире несколько собственников.

Вариант 7. Малое научно-внедренческое предприятие «Квадро» занимается прокладкой компьютерных сетей и разработкой программных комплексов для организаций. Численность работников в «Квадро» 80 человек. Одновременно находится в разработке до 30 проектов. Один разработчик может участвовать в нескольких проектах одновременно, но зарплата его от этого не зависит. Одна организация может заказать в «Квадро» несколько разработок. Стоимость каждого проекта оговаривается отдельно. При досрочном выполнении работы заказчик перечисляет научно-внедренческому предприятию определенный, заранее оговоренный процент премии.

Вариант 8. Общество с ограниченной ответственностью «Киновидеопрокат» контролирует демонстрацию кинофильмов в кинотеатрах города. Отдел маркетинга принимает решение о покупке кинолент. Отдел закупок претворяет решения в жизнь. Лента покупается у производителя и у посредника. Отдел аренды киновидеопроката сдает фильмы кинотеатрам города в аренду.

Вариант 9. Предприятие LADA-сервис занимается продажей автомобилей марки ВАЗ. Покупатель может заказать модель, цвет, тюнинг и договориться о сроках поставки автомобиля. Одновременно с новыми авто на площадках компании имеется большой выбор подержанных автомобилей отечественных и иностранных моделей.

Вариант 10. Торгово-посредническая фирма «Столица» делает закупки продуктов питания в регионах страны и за рубежом. Часть продукции закупается у местных продавцов. Фирма получает прибыль за счет того, что крупные партии товара стоят дешевле, чем мелкие. Товар не может быть продан дешевле, чем он куплен.

Методические указания

1 Создайте проект «Windows Forms Application Visual C++».
2 Разработайте диаграмму классов.
3 Формализуйте алгоритм решения задачи на ПЭВМ.
4 Используйте компоненты для работы с базой данных.
5 Выведите на экран видеомонитора результаты работы программы.
6 Оформите отчет по лабораторной работе.

1.2.3 Практическая работа №2
«Обработка баз данных в Microsoft Visual C++»

Вариант 1. В отделе кадров университета находятся данные всех сотрудников. Учитываются сведения о специальности сотрудника и занимаемой должности, ученой степени (кандидат наук, доктор) и ученом звании (доцент, профессор). В отделе кадров хранится информация о предыдущих местах работы сотрудника, сроке работы и предприятии.

Вариант 2. Разработайте программу деятельности биржи труда. Организации предоставляют бирже список свободных вакансий. Каждый обратившийся ставится на учет. В день обращения ему предлагается список вакансий. Если свободных вакансий нет или они не устраивают ищущего работу, то ему будет предложено подождать, пока подходящее свободное место работы не появится. Зарегистрированный на бирже получает пособие по безработице до тех пор, пока не будет трудоустроен.

Вариант 3. Телеателье «Спектр» занимается послегарантийным ремонтом теле-, радиоаппаратуры отечественного и импортного производства. Расчет с физическими лицами ведется наличными, а с организациями через банк. Выдача отремонтированной техники производится после оплаты выполненного ремонта. Отремонтированное изделие получает гарантию. Клиент, обратившийся к услугам телеателье несколько раз с ремонтом разной аппаратуры, получает дисконтную карту, дающую право на скидку при ремонте очередного изделия.

Вариант 4. Разработайте прикладное программное обеспечение деятельности отдела заселения муниципальных общежитий администрации города. В ведении администрации города находится несколько десятков общежитий. База данных отдела содержит информацию об общежитиях, комнатах общежитий и проживающих.

Вариант 5. Разработайте программу деятельности Государственной автомобильной инспекции по безопасности дорожного движения города. База данных ГИБДД содержит сведения обо всех транспортных средствах города и их владельцах. В нее заносятся сведения о технических осмотрах транспортных средств и об угонах. Описание угнанного автомобиля не удаляется из базы данных.

Вариант 6. Туристическая компания «Вояж» формирует туристические группы для заграничных поездок и обеспечивает им полную поддержку на маршруте. Количество туристов в группе заранее известно и ограничено. Маршрут группы может пролегать через несколько городов страны назначения. Экскурсии в несколько стран одновременно не проводятся.

Вариант 7. В собственности рекламного агентства «Rapid» находится около сотни рекламных щитов, расположенных по всему городу. Срок размещения, стоимость аренды щита и стоимость изготовления рекламы – договорные. Одна организация может арендовать несколько рекламных щитов. Один щит не сдается в аренду нескольким арендаторам, так как является

неделимой рекламной единицей. Договор размещения рекламы может быть продлен по взаимной договоренности сторон.

Вариант 8. Разработайте программу деятельности ООО «Центр оценки и продажи недвижимости». Источник прибыли организации является покупка и продажа квартир. Центр оценки имеет большой штат специалистов, позволяющий этой организации проводить сделки купли-продажи на высоком профессиональном уровне. Владелец квартиры, желающий ее продать, заключает договор, в котором указывается сумма, срок продажи и процент отчислений в случае успешного проведения сделки. Один клиент может заключить более одного договора купли-продажи одновременно, если он владеет несколькими квартирами. Обмен квартир специалисты центра непосредственно не производят. Для этих целей используется вариант купли-продажи.

Вариант 9. Отдел вневедомственной охраны квартир обеспечивает электронную охрану квартир граждан в одном районе города. Для установки охранной сигнализации требуется наличие квартирного телефона. Один гражданин может заключить договор на охрану нескольких квартир. Из-за ложных срабатываний сигнализации возможно несколько выездов патрульных экипажей по одной квартире. На владельца квартиры, вовремя не отключившего сигнализацию после своего прихода домой, налагается штраф, величина которого оговаривается при заключении договора охраны. Если отдел вневедомственной охраны не уберег имущество владельца квартиры, то он выплачивает пострадавшему заранее оговоренную сумму. От величины этой суммы зависит размер ежемесячной оплаты за охрану квартиры.

Вариант 10. Разработайте программу деятельности телефонной компании. Клиентами компании могут быть как физические лица, так и организации. Расчет с организациями ведется в безналичной форме через банк. Физические лица вносят плату через кассу компании. Клиент телефонной компании может иметь несколько телефонных номеров. Дополнительная плата за подключенный параллельно аппарат не взимается. Если телефон у абонента не работает более суток, то плата за пользование телефоном уменьшается. Междугородние и международные звонки оплачиваются отдельно по заранее установленным расценкам.

2 Основы структур данных и методы поиска

2.1 Структуры данных

2.1.1 Линейные и нелинейные структуры данных

Структура данных (data structure) – множество элементов и множество связей между ними [6, с. 10].

Структура данных – организационная схема хранения данных, в соответствии с которой данные интерпретируются и над ними выполняются операции [7, с. 133].

Линейные структуры данных – структуры, все элементы которой равноправны. К ним относятся массив, множество, список, очередь, стек, дек.

Нелинейные структуры – структуры, между элементами которых существуют отношения подчиненности или они могут быть связаны логическими условиями. К ним относятся деревья, графы, фреймы.

Линейный однонаправленный список – структура данных, состоящая из элементов одного типа, связанных между собой.

Линейные списки реализуются при помощи массивов и связных списков.

Элемент списка содержит две части: основное поле и вспомогательное поле. Основное поле содержит информацию. Например, символ. Вспомогательное поле содержит указатель на следующий элемент. Головной элемент содержит указатель Start. Последний элемент линейного списка хранит указатель NULL.

Над списками выполняются операции:

- добавление элемента в начало списка;
- добавление элемента в конец списка;
- вставка элемента в заданное место списка;
- чтение элемента с заданным ключом;
- удаление элемента из начала списка;
- удаление элемента из конца списка;
- удаление элемента с заданным ключом;
- удаление всего списка;
- упорядочивание списка по ключу;
- вывод списка.

Бинарное дерево – это динамическая структура данных, состоящая из узлов, каждый из которых содержит данные и не более двух ссылок на узлы. На каждый узел имеется ровно одна ссылка.

Корень – начальный узел дерева.

Лист – узел, не имеющий поддеревьев.

Предок – исходящий узел. Потомок – входящий узел.

Высота дерева – количеством уровней, на которых располагаются его узлы. Для бинарных деревьев определены операции:

- включение узла в дерево;
- поиск по дереву;
- обход дерева;
- удаление узла дерева.

2.1.2 Лабораторная работа №5
«Универсальная очередь ограниченного размера с использованием шаблонного класса»

Разработайте программу на языке C++, реализующую универсальную очередь ограниченного размера с использованием шаблонного класса. Программа содержит стандартные и пользовательские включаемые файлы и определение шаблонного класса для работы с универсальной очередью ограниченного размера на базе массива.

В шаблоне классов реализованы операции с очередью:

- инициализация очереди (конструктор);
- разрушение очереди с освобождением занятой динамической памяти (деструктор);
- занесение элемента с левого конца;
- занесение элемента с правого конца;
- извлечение элемента с левого конца;
- извлечение элемента с правого конца;
- печать состояния очереди с использованием указателя на левый конец очереди;
- печать состояния очереди с использованием указателя на правый конец очереди.

Вариант 1. Учет поступления счетов оплаты за потребленную электроэнергию. Плата взымается с каждой квартиры в зависимости от количества потребленной энергии или от числа, проживающих в квартире.

Вариант 2. Обслуживание покупателей билетов в кассе авиакомпании, занимающейся продажей билетов на предстоящие рейсы.

Вариант 3. Формализуйте на ЭВМ в виде очереди ограниченного размера обслуживание заявок с данными от абитуриентов, осуществляемой приемной комиссией университета.

Вариант 4. Обработка заявок на приобретение семян растений в институте селекции растений. Данный институт занимается сбором, выведением и продажей сортов семян.

Вариант 5. Оформление квитанций гостям гостиницы.

Вариант 6. Учет заявок на ремонт электровозов в депо.

Вариант 7. Обработка заявок на перевозку грузов судоходной компанией «Балтика». Компания занимается перевозками грузов между континентами.

Вариант 8. Обслуживание заявок на разработку программных комплексов научно-внедренческим предприятием.

Вариант 9. Обработка заявок на регистрацию прав юридических и физических лиц на недвижимое имущество (здания, квартиры, земельные участки).

Вариант 10. Обслуживание счетов оплаты за пользование газом и газовыми приборами предприятием «Газкомплект».

Методические указания

1 Создайте проект приложения.
2 Разработайте диаграмму классов.
3 Формализуйте алгоритм решения задачи на ПЭВМ.
4 Выведите на экран видеомонитора результаты работы программы.
5 Оформите отчет по лабораторной работе.

2.1.3 Лабораторная работа №6 «Динамический стек неограниченного размера с использованием шаблонного класса»

Разработайте программу на языке C++, реализующую динамический стек неограниченного размера с использованием шаблонного класса. Программа содержит включаемые файлы, определение структуры для элемента стека и определение шаблонного класса для работы с динамическим стеком неограниченного размера.

В шаблонном классе реализованы следующие операции со стеком:

- инициализация стека (конструктор);
- разрушение стека с освобождением занятой динамической памяти (деструктор);
- занесение элемента в стек;
- извлечение элемента из стека;
- печать состояния стека.

Вариант 1. Учет пациентов на прием к врачам поликлиники. Медицинское обслуживание работников предприятия – бесплатное (за счет средств предприятия). «Посторонние» пациенты также могут воспользоваться услугами поликлиники, полностью оплатив затраты на лечение.

Вариант 2. Обработка заявок в детские сады отделом администрации города.

Вариант 3. Обслуживание налогоплательщиков отделом учета налогообложения физических лиц городской налоговой инспекции. По существующему законодательству некоторые категории граждан должны представить в налоговую инспекцию декларацию о полученных доходах.

Вариант 4. Учет заявок на ремонт электронной аппаратуры. Радиомастерская «Электрон» занимается послегарантийным ремонтом теле-, радиоаппаратуры отечественного и импортного производства.

Вариант 5. Обработка заявок клиентов на покупку туристических путевок. Туристическая компания «Интурист» формирует туристические группы для заграничных поездок и обеспечивает им полную поддержку на маршруте.

Вариант 6. Обслуживание клиентов рекламного агентства «Rapid». В собственности этого агентства находится примерно около сотни рекламных щитов, расположенных по всему городу. Установка их согласована с

администрацией города, и все необходимые формальности выполнены. На щитах размещается реклама по заказу организации города.

Вариант 7. Обработка заявок на изготовление и выдачу технических паспортов на объекты недвижимости. Перед регистрацией сделки с объектом недвижимости собственник объекта должен получить в «Бюро технической инвентаризации» на него технический паспорт.

Вариант 8. Обслуживание заявок на установку оборудования кабельного телевидения. Клиентами компании могут быть физические лица и организации.

Вариант 9. Учет заявок на покупку книг в мелкооптовом книжном магазине. Менеджер магазина, изучив спрос на книжную продукцию в городе, принимает решение о закупке партии книг в издательстве.

Вариант 10. Обработка заявок на получение нужной информации в справочном бюро.

Методические указания

1 Создайте проект приложения.
2 Разработайте диаграмму классов.
3 Формализуйте алгоритм решения задачи на ПЭВМ.
4 Выведите на экран видеомонитора результаты работы программы.
5 Оформите отчет по лабораторной работе.

2.1.4 Практическая работа №3 «Разработка приложения, формализующего линейный список»

Вариант 1. Обработка списка налогоплательщиков отделом учета налогообложения физических лиц городской налоговой инспекции и линейный поиск. Категории граждан представляют в налоговую инспекцию декларацию о доходах.

Вариант 2. Обработка заявок на ремонт электронной аппаратуры и линейный поиск. Радиомастерская «Электрон» занимается послегарантийным ремонтом теле-, радиоаппаратуры отечественного и импортного производства.

Вариант 3. Обслуживание заявок на покупку автомобилей и поиск. Автомобиль может быть продан по предоплате.

Вариант 4. Обработка списка клиентов рекламного агентства «Rapid» и поиск. В собственности агентства находится около сотни рекламных щитов.

Вариант 5. Учет пациентов на прием к врачам поликлиники и поиск. Медицинское обслуживание работников предприятия – бесплатное (за счет средств предприятия). «Посторонние» пациенты также могут воспользоваться услугами поликлиники, полностью оплатив затраты на лечение.

Вариант 6. Обработка заявок в детские сады отделом администрации города и поиск.

Вариант 7. Обслуживание заявок на покупку книг в мелкооптовом книжном магазине и поиск. Менеджер магазина, изучив спрос на книжную продукцию в городе, принимает решение о закупке партии книг в издательстве.

Вариант 8. Обработка заявок на изготовление и выдачу технических паспортов на объекты недвижимости «Бюро технической инвентаризации». Предусмотрите поиск. Перед регистрацией сделки с объектом недвижимости собственник объекта должен получить в БТИ на него технический паспорт.

Вариант 9. Обслуживание клиентов ООО «Центр оценки и продажи недвижимости» и поиск. Источник прибыли организации – покупка и продажа квартир. Центр оценки имеет большой штат специалистов, позволяющий организации проводить сделки купли-продажи на высоком профессиональном уровне. Владелец квартиры, желающий ее продать, заключает договор с Центром, в котором указывается сумма, срок продажи и процент отчислений в пользу Центра оценки и продажи недвижимости в случае успешного проведения сделки.

Вариант 10. Обслуживание заявок клиентов на покупку туристических путевок и поиск. Туристическая компания «Интурист» формирует туристические группы для заграничных поездок и обеспечивает им полную поддержку на маршруте.

2.1.5 Практическая работа №4
«Разработка приложений, формализующих бинарное дерево поиска»

Вариант 1. Автоматизированная информационная система на автовокзале содержит сведения об отправлении пригородных автобусов. Для каждого автобуса указывается номер автобуса, пункт назначения, время отправления, время прибытия. Данные в информационной системе организованы в виде двоичного дерева. Разработайте программу, которая обеспечивает первоначальный ввод данных в информационную систему и формирование двоичного дерева, производит вывод всего дерева, вводит номер автобуса и выводит все данные об автобусе, вводит название пункта назначения и выводит данные о всех автобусах, следующих до этого населенного пункта. Программа должна обеспечивать диалог с помощью меню и контроль ошибок при вводе.

Вариант 2. Разработайте программу, которая содержит информацию о сотрудниках, работающих в фирме. Сведения о сотрудниках содержат табельный номер, фамилию, имя, отчество, образование, год поступления на работу, домашний адрес, оклад. Программа должна обеспечивать начальное формирование данных обо всех сотрудниках фирмы в виде двоичного дерева, добавление данных о сотрудниках, вновь принятых на работу, удаление данных о сотрудниках, уволенных с работы, по запросу выдавать сведения о сотрудниках в штате фирмы, упорядоченные по фамилии, имени, отчеству.

Вариант 3. Разработайте программу, которая содержит информацию о моделях компьютеров, продаваемых в магазине вычислительной техники.

Сведения о компьютере содержат марку компьютера, тип процессора, тактовую частоту процессора, объем памяти, объем жесткого диска, объем памяти видеокарты, цену компьютера, количество экземпляров, имеющихся в наличии. Программа должна обеспечивать начальное формирование данных обо всех компьютерах в магазине вычислительной техники в виде двоичного дерева, добавление данных о компьютерах, поступающих в магазин, удаление данных о проданных компьютерах, выдавать сведения о наличии компьютеров в магазине, упорядоченные по наименованию модели.

Вариант 4. Разработайте программу, которая содержит информацию о реестре жилых помещений (купля/продажа) фирмы. Данные реестра жилья содержат район, адрес, количество комнат, общую площадь, жилую площадь, год постройки дома, стоимость. Программа должна обеспечивать хранение всех данных о жилых помещениях в виде двоичного дерева, добавление в реестр данных о жилых помещениях, удаление данных о проданных жилых помещениях из реестра фирмы, вывод данных о жилых помещениях по стоимости, вывод всех жилых помещений, занесенных в реестр.

Вариант 5. Разработайте программу, которая содержит информацию о дисциплинах, читаемых преподавателем студентам университета в течение учебного года. Сведения о нагрузке преподавателя за учебный год содержат название дисциплины, семестр проведения занятия, количество студентов, количество часов аудиторных лекций, количество часов аудиторных практических работ, вид контроля знаний студентов (зачет/экзамен). Программа должна обеспечивать начальное формирование данных о дисциплинах, читаемых преподавателем, в виде двоичного дерева, добавление данных о дисциплинах, удаление данных о дисциплинах, вывод данных о дисциплинах по наименованию, вывод всех дисциплин, составляющих нагрузку преподавателя.

Вариант 6. Разработайте программу, которая содержит информацию о дилерах компании. Сведения о дилерах содержат адрес, фамилию, имя, отчество, телефон, электронный адрес, объем закупок продукции в месяц, объем продаж продукции за месяц, льготный процент скидки при закупке продукции. Программа должна обеспечивать начальное формирование данных обо всех дилерах фирмы в виде двоичного дерева, добавление данных о дилерах, удаление данных о дилерах, выдавать сведения о дилерах по фамилии, имени, отчеству.

Вариант 7. Разработайте программу, которая содержит информацию о безработных, зарегистрированных на бирже труда. Данные о зарегистрированных безработных содержат номер регистрации безработного, фамилию, имя, отчество, возраст, пол, образование, профессию, общий стаж работы, дату постановки на учет, желаемую заработную плату, должность. Программа должна обеспечивать хранение всех зарегистрированных безработных в виде двоичного дерева, добавление данных о безработных, удаление данных о безработных, нашедших работу, вывод данных о

безработных по фамилии, имени, отчеству, регистрационному номеру, вывод всех зарегистрированных безработных.

Вариант 8. Автоматизированная информационная система на железнодорожном вокзале содержит сведения об отправлении поездов дальнего следования. Для каждого поезда указывается номер поезда, станция назначения, время отправления. Данные в информационной системе организованы в виде двоичного дерева. Разработайте программу, которая обеспечивает первоначальный ввод данных в информационную систему и формирование двоичного дерева, производит вывод всего дерева, вводит номер поезда и выводит все данные об этом поезде, вводит название станции назначения и выводит данные о всех поездах, следующих до этой станции.

Вариант 9. На междугородной телефонной станции картотека абонентов, содержащая сведения о телефонах и их владельцах, организована как двоичное дерево. Разработайте программу, которая обеспечивает начальное формирование картотеки в виде двоичного дерева, производит вывод всей картотеки, вводит номер телефона и время разговора, выводит извещение на оплату телефонного разговора.

Вариант 10. Англо-русский словарь построен как двоичное дерево. Каждая компонента содержит английское слово, соответствующее ему русское слово и счетчик количества обращений к данной компоненте. Первоначально дерево формируется согласно английскому алфавиту. В процессе эксплуатации словаря при каждом обращении к компоненте в счетчик обращений добавляется единица. Разработайте программу, которая обеспечивает начальный ввод словаря с конкретными значениями счетчиков обращений, обеспечивает вывод исходного и нового словарей, формирует новое представление словаря в виде двоичного дерева по алгоритму:

1) в старом словаре ищется компонента с наибольшим значением счетчика обращений;

2) найденная компонента заносится в новый словарь и удаляется из старого;

3) переход к этапу 1 до окончания исходного словаря.

2.2 Поиск данных

2.2.1 Алгоритмы поиска данных

Алгоритм поиска – алгоритм, который в заданном файле ищет запись с заданным ключом [7, с. 244].

Алгоритмы поиска подразделяются на:

- внешние и внутренние;
- статические и динамические;
- основанные на сравнении ключей или на цифровых свойствах ключей;
- использующие ключи или образы [7, с. 245].

Линейный поиск – алгоритм поиска, заключающийся в последовательном определении элемента (слева направо), совпадающего с ключом поиска. Применяется для небольших несортированных массивов. Для большого массива линейный поиск неэффективен.

Бинарный (дихотомический) поиск применяется для отсортированной последовательности элементов, к которым можно получить прямой доступ посредством индекса.

Интерполяционный поиск производит определение местонахождения элемента. Множество хранится в отсортированном массиве.

Метод хеширования использует при поиске целочисленный образ ключа элемента. Данный метод позволяет сократить вычислительную сложность поиска при возможном увеличении памяти [7, с. 313].

2.2.2 Лабораторная работа №7 «Поиск данных в таблице»

Вариант 1. ОАО «Аэровокзал». Это открытое акционерное общество занимается междугородними пассажирскими авиаперевозками по России. В его собственности находится несколько десятков авиалайнеров различной вместимости. Билеты на рейсы продаются только в здании аэровокзала. Возможна предварительная продажа. Маршрут авиалайнера может пролегать через несколько населенных пунктов. В этом случае пассажир может купить билет до любого промежуточного пункта.

Вариант 2. Мелкооптовый книжный магазин. Менеджер магазина, изучив спрос на книжную продукцию в городе, принимает решение о закупке партии книг в том или ином издательстве. Некоторые пользующиеся повышенным спросом книги могут быть закуплены у посредников. Часть продукции заказывается через интернет. Покупателем в мелкооптовом магазине может быть любой человек или организация при условии, что величина покупки превысит одну тысячу рублей. Расчет с организациями производится через банк. Расчет с физическими лицами – наличными.

Вариант 3. Телефонная компания «Bell Line». Основное назначение приложения – отслеживание абонентской платы за телефоны. Клиентами компании могут быть как физические лица, так и организации. Расчет с организациями ведется в безналичной форме через банк. Физические лица вносят плату через кассу компании. Клиент телефонной компании может иметь несколько телефонных номеров.

Вариант 4. Туристическая компания «Вояж». Эта компания формирует туристические группы для заграничных поездок и обеспечивает им полную поддержку на маршруте. Количество туристов в группе заранее известно и ограничено. Маршрут группы может пролегать через несколько городов страны назначения.

Вариант 5. Компания кабельного телевидения. Компания предоставляет пакеты программ кабельного телевидения абонентам и производит учет оплаты за услуги. Постоянным клиентам предоставляются скидки.

Вариант 6. Компания спутниковой телефонной связи. Компания предоставляет услуги спутниковой телефонии, производит продажу спутникового оборудования и учет абонентской платы пользователей спутниковой телефонной связи.

Вариант 7. Компания продажи автомобилей. Компания продает новые автомобили ГАЗ, производит сервисное обслуживание и ремонт, учет покупателей автомобилей.

Вариант 8. Сотрудники фирмы. Сведения о сотрудниках содержат табельный номер, фамилию, имя, отчество, образование, год поступления на работу, домашний адрес, оклад. Программа должна обеспечивать начальное формирование данных обо всех сотрудниках фирмы в виде двоичного дерева, добавление данных о сотрудниках, вновь принятых на работу, удаление данных о сотрудниках, уволенных с работы, по запросу выдавать сведения о сотрудниках в штате фирмы, упорядоченные по фамилии, имени, отчеству.

Вариант 9. Магазин вычислительной техники. Сведения о компьютере содержат марку компьютера, тип процессора, тактовую частоту процессора, объем памяти, объем жесткого диска, объем памяти видеокарты, цену компьютера, количество экземпляров, имеющихся в наличии. Программа должна обеспечивать начальное формирование данных обо всех компьютерах в магазине вычислительной техники в виде двоичного дерева, добавление данных о компьютерах, поступающих в магазин, удаление данных о проданных компьютерах, выдавать сведения о наличии компьютеров в магазине, упорядоченные по наименованию модели.

Вариант 10. Дилеры компании. Сведения о дилерах содержат адрес, фамилию, имя, отчество, телефон, электронный адрес, объем закупок продукции в месяц, объем продаж продукции за месяц, льготный процент скидки при закупке продукции. Программа должна обеспечивать начальное формирование данных обо всех дилерах фирмы в виде двоичного дерева, добавление данных о дилерах, удаление данных о дилерах, выдавать сведения о дилерах по фамилии, имени, отчеству.

Методические указания

1 Создайте проект «Windows Forms Application Visual C++».
2 Разработайте диаграмму классов.
3 Формализуйте алгоритм решения задачи на ПЭВМ.
4 Выведите на экран видеомонитора результаты работы программы.
5 Оформите отчет по лабораторной работе.

3 Реализация методами объектно-ориентированного программирования сверхдлинной целочисленной арифметики

3.1 Операции со сверхдлинными целыми числами

3.1.1 Аддитивные и мультипликативные операции

Длинная арифметика – операции (сложение, умножение, вычитание, деление, возведение в степень и т. д.) над числами, разрядность которых превышает длину машинного слова вычислительной машины.

Представление целого числа a в системе счисления с основанием B имеет вид

$a = an\text{-}1Bn\text{-}1 + an\text{-}2Bn\text{-}2 + \ldots + a1B + a0,$

где $0 \le ai < B$.

Данное представление целого числа аналогично представлению полинома степени n - 1:

$a(x) = an\text{-}1xn\text{-}1 + an\text{-}2xn\text{-}2 + \ldots + a1x + a0$

с коэффициентами ai, $0 \le ai < B$.

Сверхдлинное целое число хранится в виде массива цифр. Цифры используются из системы счисления. Применяется десятичная система счисления и её степени (десять тысяч, миллиард) либо двоичная система счисления.

Операции над числами длинной арифметики производятся с помощью «школьных» алгоритмов сложения, вычитания, умножения, деления столбиком.

Рассмотрим два целых числа a, b по некоторому основанию B, т. е.

$a = an\text{-}1Bn\text{-}1 + an\text{-}2Bn\text{-}2 + \ldots + a1B + a0,$

$b = bn\text{-}1Bn\text{-}1 + bn\text{-}2Bn\text{-}2 + \ldots + b1B + b0,$

где $0 \le ai < B$, $0 \le bi < B$.

Для сложения чисел используется следующий алгоритм.

Алгоритм 1. Найти сумму $w = a + b$.

1 Определить $c = 0$, $i = 0$.

2 Для $i = 0, 1, \ldots, n\text{-}1$ выполнить шаги 3-5 алгоритма.

3 Вычислить $t = ai + bi + c$.

4 Положить $c = t \bmod B$.

5 $wi = c$.

6 Положить $wn = t$.

7 Результат $w = wnBn + wn\text{-}1Bn\text{-}1 + wn\text{-}2Bn\text{-}2 + \ldots + w1B + w0.$

Для вычитания чисел используется следующий алгоритм.

Алгоритм 2. Найти разность $w = a – b$, где $0 < b \le a$.

1 Определить $c = 0$, $i = 0$.

2 Для $i = 0, 1, \ldots, n\text{-}1$ выполнить шаги 3-5 алгоритма.

3 Вычислить $t = ai - bi + c$.

4 Положить $c = t \bmod B$.

5 $wi = c$.

6 Результат $w = wn\text{-}1Bn\text{-}1 + wn+m\text{-}2Bn+m\text{-}2 +...+ w1B + w0$.

Заметим, что в процессе выполнения алгоритма, переменная c может принимать отрицательное значение, т. е. может выполняться неравенство $c < 0$.

Алгоритм 3. Найти произведение двух чисел a (n – разрядное число) и b(m – разрядное число): $w = a * b$.

1 Для $i = 0, 1, \ldots, n\text{-}1$.

2 Определить $wi = 0$.

3 Для $i = 0, 1, \ldots, m\text{-}1$ (выполнить шаги 4-9 алгоритма).

4 Определить $c = 0$.

5 Для $j = 0, 1, \ldots, n\text{-}1$ (выполнить шаги 6-8 алгоритма).

6 Вычислить $t = wi+j + ai \cdot bj + c$.

7 Положить $wi+j = t \bmod B$.

8 $c = t \bmod B$.

9 $wi+n = c$.

10 Результат $wn+m\text{-}1Bn+m\text{-}1 + wn+m\text{-}2Bn+m\text{-}2 +...+ w1B + w0$.

Рассмотрим алгоритм деления «в столбик».

Алгоритм 4. Найти результат деления двух чисел a ($n + m$ – разрядное число) и b (n – разрядное число). Начальные данные алгоритма: целые числа $a = (am+n\text{-}1,\ldots, a0)B$, $b = (bn\text{-}1 ,\ldots, b0)B$, где B – основание системы счисления.

Результат выполнения алгоритма: целые числа $q = (qm, qm\text{-}1 ,\ldots, q0)B$ (частное от деления двух чисел) и $r = (rn, rn\text{-}1 ,\ldots, r0)B$ (остаток от деления двух чисел) такие, что $a = qb + r$, $0 \le r < b$.

1 Определить такое целое число $d > 0$, что $d \cdot bn\text{-}1 \ge [B/2]$.

2 Положить $(rm+n , rm+n\text{-}1 ,\ldots, r0)B \leftarrow (0, am+n\text{-}1, am+n\text{-}2 ,\ldots, a0)B$.

3 Для $i = m + n, m + n\text{-} 1, ..., n$ выполнить следующие действия (шаги 4-7).

4 Определить $Q = \min([(Ri \cdot B + Ri\text{-}1)/Vn\text{-}1], B - 1)$, где $Ri = ri \cdot d$, $Vn\text{-}1 = bn\text{-}1 \cdot d$, а выражение в квадратных скобках означает целая часть числа.

5 Пока $Vn\text{-}2 \cdot Q > (RiB + Ri\text{-}1 – QVn\text{-}1)B + Ri\text{-}2$ выполнять $Q = Q – 1$.

6 Вычислить $w = r – bQ$ и, если $w < 0$, положить $Q = Q – 1$.

7 Положить $w = r – bQ$ и $qi\text{-}n = Q$.

8 Результат: $q = (qm , qm\text{-}1 ,\ldots, q0)B$ **,** $r = (rn , rn\text{-}1 ,\ldots, r0)B$.

3.1.2 Лабораторная работа №8
«Калькулятор сверхдлинных целых чисел»

Вариант 1-10. Разработайте программу на языке C++, используя визуальное программирование. Программа формализует работу калькулятора сверхдлинных чисел. Калькулятор позволяет выполнять операции:

1 Аддитивные и мультипликативные операции:

- сложения +;
- вычитания -;
- умножения *;
- деления /.

2 Операции сравнения сверхдлинных чисел:

- равно ==;
- меньше <;
- больше >;
- меньше или равно <=;
- больше или равно >=.

3 Логические и побитовые операции:

- побитовое И(&);
- побитовое исключающее ИЛИ (^);
- побитовое ИЛИ (|);
- логическое ИЛИ (||);
- логическое И (&&).

4 Определение наибольшего общего делителя (НОД) и наименьшего общего кратного (НОК).

Методические указания

1 Создайте проект «Windows Forms Application Visual C++».
2 Разработайте диаграмму классов.
3 Формализуйте алгоритм решения задачи на ПЭВМ.
4 Выведите на экран видеомонитора результаты работы программы.
5 Оформите отчет по лабораторной работе.

4 Элементы библиотеки стандартных шаблонов

4.1 Контейнеры стандартной библиотеки

Стандартная библиотека шаблонов STL (Standard Template Library) – библиотека контейнерных классов, включающая векторы, списки, очереди, стеки и алгоритмы общего назначения.

4.1.1 Последовательные контейнеры

Последовательный контейнер – вид контейнеров, в котором введено отношение порядка для совокупности хранимых объектов. Для элементов контейнера определены понятия первый, последний, предыдущий и следующий.

Последовательные контейнеры обеспечивают хранение конечного количества однотипных объектов в виде последовательности и имеют разновидности: векторы (vector), деки (deque), списки (list), стеки (stack), очереди (queue), очереди с приоритетом (priority_queue). Первые три последовательных контейнеров являются основными, остальные вспомогательными контейнерами [8, с. 353].

4.1.2 Лабораторная работа №9
«Разработка приложений с использованием последовательных контейнеров»

Разработайте программу на языке C++, используя стандартную библиотеку шаблонов.

Вариант 1. Последовательный контейнер вектор, его конструкторы и характеристики.

Разработайте законченную программу, в которой с помощью конструктора умолчания, различных разновидностей обычного конструктора и конструктора копирования создаются объекты-вектора. Выведите на экран для контроля значения некоторых из элементов созданных векторов, размеры созданных векторов и количество элементов векторов, которое может храниться без перераспределения памяти.

Используйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод, класс *vector*).

Вариант 2. Последовательный контейнер вектор. Присваивание векторов.

Разработайте законченную программу, в которой с помощью подходящих конструкторов создайте три вектора *v1*, *v2*, *v3* с элементами целого типа, размерами соответственно 5, 7, 6 и одинаковыми значениями элементов соответственно 1, 2, 3. Выведите на экран размеры векторов, значения их элементов и выполните присваивание *v3=v2-v1*. После этого вновь выведите на экран размеры векторов и значения их элементов.

Используйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод, класс *vector*).

Вариант 3. Последовательный контейнер вектор. Копирование векторов.

Разработайте законченную программу, в которой с помощью подходящих конструкторов создайте три вектора *v1*, *v2*, *v3* с элементами целого типа, размерами соответственно 4, 5, 7 и одинаковыми значениями элементов соответственно 1, 2, 3. Выведите на экран размеры векторов, значения их элементов. С помощью метода *assign( )* первым трем элементам *v1* присвойте значение 4, а первым двум элементам *v2* присвойте значения элементов *v3[4]* и *v3[5]*. После этого вновь выведите на экран размеры векторов и значения их элементов.

Используйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод, класс *vector*).

Вариант 4. Последовательный контейнер вектор. Доступ к элементам вектора.

Разработайте законченную программу, в которой с помощью подходящего конструктора создайте вектор *v* размером 5 с элементами целого типа, равными 1. Выведите на экран размер вектора и значения его элементов. С помощью операции [] второй элемент вектора увеличьте на 3, а с помощью метода *at( )* четвертый элемент вектора уменьшите на 1. Вновь выведите на экран размер вектора и значения его элементов. С помощью методов *front()* и

back( ) первый элемент вектора уменьшите на 5, а последний – увеличьте на два.

Используйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод, класс *vector*).

Вариант 5. Последовательный контейнер вектор. Резервирование памяти под вектор и изменение его размера.

Разработайте законченную программу, в которой с помощью подходящих конструкторов создайте пустой вектор *v1* с элементами целого типа и вектор *v2* размером 4 и значениями элементов 2. С помощью метода *reserve( )* зарезервируйте память под вектор *v1* до 15, напечатайте на экране для этого вектора фактический размер и зарезервированный размер. С помощью метода *resize( )* измените размер вектора *v2* до шести, добавленные элементы инициализируйте значением 10, выведите на экран размер вектора *v2* и значения его элементов.

Используйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод, класс *vector*).

Вариант 6. Последовательный контейнер вектор. Изменение вектора.

Разработайте законченную программу, в которой с помощью подходящих конструкторов создайте два вектора *v1, v2* с размерами 5 и элементами целого типа, имеющими одинаковые значения 1 и –2 соответственно. Выведите на экран размеры и значения элементов созданных векторов. С помощью метода *push_back( )* добавьте элемент со значением 21 в конец вектора *v1* и выведите на экран размер и значения элементов вектора *v1*. С помощью метода *pop_back( )* удалите последний элемент вектора *v1* и выведите на экран размер и значения элементов вектора *v1*.

Используйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод, класс *vector*).

Вариант 7. Последовательный контейнер вектор. Изменение вектора.

Разработайте законченную программу, в которой с помощью подходящих конструкторов создайте два вектора *v1, v2* с размерами 5 и элементами целого типа, имеющими одинаковые значения 1 и –2 соответственно. Выведите на экран размеры и значения элементов созданных векторов. С помощью метода *insert( )* вставьте элемент со значением 12 после третьего элемента вектора *v1* и выведите на экран размер и значения элементов вектора *v1*. С помощью метода *insert( )* вставьте два элемента со значением 13 после четвертого элемента вектора *v1* и выведите на экран размер и значения элементов вектора *v1*. С помощью метода *insert( )* вставьте в начало вектора *v2* третий и четвертый элементы вектора *v1* и выведите на экран размер и значения элементов вектора *v2*.

Используйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод, класс *vector*).

Вариант 8. Последовательный контейнер вектор. Изменение вектора.

Разработайте законченную программу, в которой с помощью подходящих конструкторов создайте два вектора *v1, v2* с размерами 5 и элементами целого

типа, имеющими одинаковые значения 1 и –2 соответственно. Выведите на экран размеры и значения элементов созданных векторов. С помощью метода *erase( )* удалите второй элемент вектора *v1* и выведите на экран размер и значения элементов вектора *v1*. С помощью метода *erase( )* удалите первый и второй элементы вектора *v2* и выведите на экран размер и значения элементов вектора *v2*. С помощью метода *clear( )* очистите вектор *v1* и выведите на экран его размер и значения элементов вектора *v1*. С помощью метода *swap( )* выполните обмен векторов *v1, v2* и выведите на экран размеры и значения элементов этих векторов.

Используйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод, класс *vector*).

Вариант 9. Последовательный контейнер вектор. Операции отношений над векторами.

Разработайте законченную программу, в которой с помощью подходящих конструкторов создайте вектора *v1, v2, v3, v4, v5* с элементами целого типа и размерами соответственно 5, 5, 6, 5, 6 и одинаковыми значениями элементов соответственно 2, 2, 2, -2, -2. Выведите на экран размеры и значения элементов созданных векторов. Выведите на экран результаты следующих сравнений векторов: *v1== v2, v1== v3, v1< v3, v1!= v3, v1<= v3, v3>= v1, v3> v1, v5>= v4.*

Используйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод, класс *vector*).

Вариант 10. Последовательный контейнер – вектор булевских значений.

Разработайте законченную программу, в которой с помощью подходящего конструктора создайте вектор булевых значений *v* размером 5 с одинаковыми значениями элементов *true*. Выведите на экран размер и значения элементов созданного вектора с использованием итераторов. Введите с клавиатуры значения элементов вектора *v* с использованием операции []. Предусмотрите обработку возможных ошибок ввода. Выведите на экран значения элементов созданного вектора с использованием операции [].

Используйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод, класс *vector*).

Методические указания

1 Создайте проект «Windows Forms Application Visual C++».

2 Разработайте диаграмму классов.

3 Формализуйте алгоритм решения задачи на ПЭВМ.

4 Выведите на экран видеомонитора результаты работы программы.

5 Оформите отчет по лабораторной работе.

4.1.3 Практическая работа №6 «Последовательные контейнеры»

Вариант 1. Последовательный контейнер – список.

Разработайте законченную программу, в которой создайте и инициализируйте списки *L1*, *L2* и выведите информацию о них на экран. С помощью метода *splice( )* вставьте список *L2* перед вторым элементом списка *L1* и переместите последний элемент списка *L1* в его начало. Выведите на экран размер списка *L1* и его состояние.

Используйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод, класс *list*).

Вариант 2. Последовательный контейнер – список.

Разработайте законченную программу, в которой создайте и инициализируйте списки *L1*, *L2* и выведите информацию о них на экран. С помощью метода *remove( )* удалите из списка *L2* элемент со значением 12 и выведите на экран информацию о списке *L2*. С помощью метода *merge( )* выполните слияние в список *L1* списков *L1*, *L2* и выведите информацию о списке *L1* на экран. С помощью методов *sort( )* и *reverse( )* последовательно вначале отсортируйте список *L1* по возрастанию, а затем измените порядок следования элементов на противоположный. После каждой операции выведите информацию о списке *L1* на экран.

Используйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод, класс *list*).

Вариант 3. Стек (*stack*).

Разработайте законченную программу, в которой создайте на базе последовательного контейнера *vector* три пустых стека *s1, s2, s3* для хранения целых значений. Стек *s1* инициализируйте значениями из файла *stack.cpp,* стек *s2* – значениями 0, 1, 2, ..., а стек *s3* – значениями 1, 2, 3, ... С помощью методов *empty( )*, *top( )* и *pop( )* выведите содержимое стека *s1* на экран, после чего стек окажется пустым. Сравните стеки *s2, s3* и результат сравнения выведите на экран. С помощью методов *empty( )*, *top( )* и *pop( )* выведите содержимое стеков *s2, s3* на экран, после чего стеки окажутся пустыми.

Используйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод, классы *vector, stack*).

Вариант 4. Стек (*stack*).

Разработайте законченную программу, в которой создайте на базе последовательного контейнера *deque* три пустых стека *s1*, *s2, s3* для хранения целых значений. Стек *s1* инициализируйте значениями из файла *stack.cpp,* стек *s2* – значениями 0, 1, 2, ..., а стек *s3* – значениями 1, 2, 3, ... С помощью методов *empty( )*, *top( )* и *pop( )* выведите содержимое стека *s1* на экран, после чего стек окажется пустым. Сравните стеки *s2, s3* и результат сравнения выведите на экран. С помощью методов *empty( )*, *top( )* и *pop( )* выведите содержимое стеков *s2, s3* на экран, после чего стеки окажутся пустыми.

Используйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод, классы *deque, stack*).

Вариант 5. Стек (*stack*).

Разработайте законченную программу, в которой создайте на базе последовательного контейнера *list* три пустых стека *s1*, *s2, s3* для хранения целых значений. Стек *s1* инициализируйте значениями из файла *stack.cpp,* стек *s2* – значениями 0, 1, 2, ..., а стек *s3* – значениями 1, 2, 3, ... С помощью методов *empty( )*, *top( )* и *pop( )* выведите содержимое стека *s1* на экран, после чего стек окажется пустым. Сравните стеки *s2, s3* и результат сравнения выведите на экран. С помощью методов *empty()*, *top( )* и *pop( )* выведите содержимое стеков *s2, s3* на экран, после чего стеки окажутся пустыми.

Используйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод, классы *list, stack*).

Вариант 6. Очередь *FIFO* (*queue)*.

Разработайте законченную программу, в которой создайте на базе последовательного контейнера *list* три пустых адаптера *queue q1*, *q2, q3* для хранения целых значений. Адаптер *q1* инициализируйте значениями из файла *queue.cpp,* адаптер *q2* – значениями 0, 1, 2, ..., а адаптер *q3* – значениями 1, 2, 3, ... С помощью методов *empty( )*, *front( )* и *pop( )* выведите содержимое адаптера *q1* на экран, после чего адаптер окажется пустым. Сравните адаптеры *q2*, *q3* и результат сравнения выведите на экран. С помощью методов *empty()*, *front( )* и *pop( )* выведите содержимое адаптеров *q2*, *q3* на экран, после чего адаптеры окажутся пустыми.

Используйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод, классы *list, queue*).

Вариант 7. Очередь *FIFO* (*queue)*.

Разработайте законченную программу, в которой создайте на базе последовательного контейнера *deque* три пустых адаптера *queue q1*, *q2, q3* для хранения целых значений. Адаптер *q1* инициализируйте значениями из файла *queue.cpp,* адаптер *q2* – значениями 0, 1, 2, ..., а адаптер *q3* – значениями 1, 2, 3, ... С помощью методов *empty( )*, *front( )* и *pop( )* выведите содержимое адаптера *q1* на экран, после чего адаптер окажется пустым. Сравните адаптеры *q2*, *q3* и результат сравнения выведите на экран. С помощью методов *empty()*, *front( )* и *pop( )* выведите содержимое адаптеров *q2*, *q3* на экран, после чего адаптеры окажутся пустыми.

Используйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод, классы *deque, queue*).

Вариант 8. Очередь с приоритетами (*priority_queue)*.

Разработайте законченную программу, в которой создайте на базе последовательного контейнера *vector* пустую очередь с приоритетами *pq* для хранения целых значений по убыванию. Очередь с приоритетами *pq* инициализируйте произвольными значениями. С помощью методов *empty( )*, *top( )* и *pop( )* выведите содержимое очереди *pq* на экран.

Используйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод, классы *vector, queue, functional*).

Вариант 9. Очередь с приоритетами (*priority_queue)*.

Напишите законченную программу, в которой создайте на базе последовательного контейнера *deque* пустую очередь с приоритетами *pq* для хранения целых значений по убыванию. Очередь с приоритетами *pq* инициализируйте произвольными значениями. С помощью методов *empty( )*, *top( )* и *pop( )* выведите содержимое очереди *pq* на экран.

Используйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод, классы *deque, queue, functional*).

Вариант 10. Последовательный контейнер – список.

Разработайте законченную программу, в которой с помощью конструктора умолчания создайте три пустых списка *L1*, *L2*, *L3* из элементов целого типа. В начало первого списка занесите значения 0, 1, 2, 3, 4, в конец второго списка – значения 10, 11, 12 и выполните присваивание *L3=L2*. С помощью итераторов выведите на экран размеры и значения элементов созданных списков.

Используйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод, класс *list*).

4.1.4 Ассоциативные контейнеры

Ассоциативные контейнеры обеспечивают быстрый доступ к данным. Они построены на основе сбалансированных деревьев поиска. Существуют следующие типы ассоциативных контейнеров:

- словари (map);
- словари с дубликатами (multimap);
- множества (set);
- множества с дубликатами (multiset);
- битовые множества (bitset).

Словарь построен на основе пар «ключ/значение». Ключ ассоциирован с элементом. В качестве ключа используется значение произвольного типа. Ключ имеет уникальное значение [8, с. 406].

4.1.5 Лабораторная работа №10 «Разработка приложений с использованием ассоциативных контейнеров»

Разработайте программу на языке C++, используя стандартную библиотеку шаблонов.

Вариант 1. Ассоциативные контейнеры. Словари (*map)*.

Напишите законченную программу, в которой создайте пустой словарь – телефонную книгу для хранения записей в лексикографическом порядке. Заполните телефонную книгу данными из файла *map.dat*. Каждая строка файла хранит фамилию абонента и телефонный номер-число, разделенные пробелом. Выведите на экран содержимое телефонной книги. Дополните телефонную

книгу новой записью и измените один из номеров телефонной книги. Снова выведите на экран содержимое телефонной книги. Выполните поиск в словаре существующего и несуществующего элемента и выведите результат поиска на экран.

Используйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод, классы *string, map*).

Вариант 2. Ассоциативные контейнеры. Словари (*map)*.

Напишите законченную программу, в которой создайте две пустых телефонных книги для хранения записей в лексикографическом порядке. Заполните телефонные книги данными из файла *map1.dat* и *map2.dat*. Каждая строка файла хранит фамилию абонента и телефонный номер-число, разделенные пробелом. Выведите на экран содержимое телефонных книг. Выполните обмен словарей и выведите их на экран. Очистите словари.

Используйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод, классы *string, map*).

Вариант 3. Ассоциативные контейнеры. Словари с дубликатами (*multimap)*.

Напишите законченную программу, в которой создайте пустую телефонную книгу – словарь с дубликатами – для хранения записей в лексикографическом порядке (фамилия абонента – ключ, число – значение, номер телефона). С помощью метода *insert( )* заполните телефонную книгу данными так, чтобы телефонная книга содержала дубликаты с одинаковыми номерами. Выведите на экран содержимое телефонной книги. Выполните поиск в словаре информации с заданными ключами и выведите результаты поиска на экран (ключ с дубликатами, без дубликатов и отсутствующий ключ).

Используйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод, классы *string, map*).

Вариант 4. Ассоциативные контейнеры. Множества (*set)*.

Напишите законченную программу, в которой с помощью подходящего конструктора создайте множество и инициализируйте его четырьмя строковыми значениями. С помощью итераторов выведите содержимое созданного множества на экран. Добавьте в множество еще два строковых значения, одно из которых уже имеется в множестве. Еще раз выведите содержимое множества на экран.

Используйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод, классы *string, set*).

Вариант 5. Ассоциативные контейнеры. Множества с дубликатами (*multiset)*.

Напишите законченную программу, в которой с помощью подходящего конструктора создайте множество с дубликатами и инициализируйте его четырьмя строковыми значениями. С помощью итераторов выведите содержимое созданного множества на экран. Добавьте в множество еще два строковых значения, одно из которых уже имеется в множестве. Еще раз выведите содержимое множества на экран.

Используйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод, классы *string, set*).

Вариант 6. Ассоциативные контейнеры. Словари (*map)*.

Напишите законченную программу, в которой создайте пустой словарь – телефонную книгу для хранения записей в лексикографическом порядке. Заполните телефонную книгу данными из файла *map.dat*. Каждая строка файла хранит фамилию абонента и телефонный номер-число, разделенные пробелом. Выведите на экран содержимое телефонной книги. Дополните телефонную книгу новой записью и измените один из номеров телефонной книги. Снова выведите на экран содержимое телефонной книги. Выполните поиск в словаре существующего и несуществующего элемента и выведите результат поиска на экран.

Используйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод, классы *string, map*).

Вариант 7. Ассоциативные контейнеры. Словари (*map)*.

Напишите законченную программу, в которой создайте две пустых телефонных книги для хранения записей в лексикографическом порядке. Заполните телефонные книги данными из файла *map1.dat* и *map2.dat*. Каждая строка файла хранит фамилию абонента и телефонный номер-число, разделенные пробелом. Выведите на экран содержимое телефонных книг. Выполните обмен словарей и выведите их на экран. Очистите словари.

Используйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод, классы *string, map*).

Вариант 8. Ассоциативные контейнеры. Словари с дубликатами (*multimap)*.

Напишите законченную программу, в которой создайте пустую телефонную книгу – словарь с дубликатами – для хранения записей в лексикографическом порядке (фамилия абонента – ключ, число – значение, номер телефона). С помощью метода *insert( )* заполните телефонную книгу данными так, чтобы телефонная книга содержала дубликаты с одинаковыми номерами. Выведите на экран содержимое телефонной книги. Выполните поиск в словаре информации с заданными ключами и выведите результаты поиска на экран (ключ с дубликатами, без дубликатов и отсутствующий ключ).

Используйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод, классы *string, map*).

Вариант 9. Ассоциативные контейнеры. Множества (*set)*.

Напишите законченную программу, в которой с помощью подходящего конструктора создайте множество и инициализируйте его четырьмя строковыми значениями. С помощью итераторов выведите содержимое созданного множества на экран. Добавьте в множество еще два строковых значения, одно из которых уже имеется в множестве. Еще раз выведите содержимое множества на экран.

Используйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод, классы *string, set*).

Вариант 10. Ассоциативные контейнеры. Множества с дубликатами (*multiset)*.

Напишите законченную программу, в которой с помощью подходящего конструктора создайте множество с дубликатами и инициализируйте его четырьмя строковыми значениями. С помощью итераторов выведите содержимое созданного множества на экран. Добавьте в множество еще два строковых значения, одно из которых уже имеется в множестве. Еще раз выведите содержимое множества на экран.

Используйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод, классы *string, set*).

Методические указания

1 Создайте проект «Windows Forms Application Visual C++».
2 Разработайте диаграмму классов.
3 Формализуйте алгоритм решения задачи на ПЭВМ.
4 Выведите на экран видеомонитора результаты работы программы.
5 Оформите отчет по лабораторной работе.

4.1.6 Практическая работа №7 «Ассоциативные контейнеры»

Вариант 1. Ассоциативные контейнеры. Словари с дубликатами (*multimap)*.

Напишите законченную программу, в которой создайте пустую телефонную книгу – словарь с дубликатами – для хранения записей в лексикографическом порядке (фамилия абонента – ключ, число – значение, номер телефона). С помощью метода *insert( )* заполните телефонную книгу данными так, чтобы телефонная книга содержала дубликаты с одинаковыми номерами. Выведите на экран содержимое телефонной книги. Выполните поиск в словаре информации с заданными ключами и выведите результаты поиска на экран (ключ с дубликатами, без дубликатов и отсутствующий ключ).

Используйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод, классы *string, map*).

Вариант 2. Ассоциативные контейнеры. Множества (*set)*.

Напишите законченную программу, в которой с помощью подходящего конструктора создайте множество и инициализируйте его четырьмя строковыми значениями. С помощью итераторов выведите содержимое созданного множества на экран. Добавьте в множество еще два строковых значения, одно из которых уже имеется в множестве. Еще раз выведите содержимое множества на экран.

Используйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод, классы *string, set*).

Вариант 3. Ассоциативные контейнеры. Множества с дубликатами (*multiset)*.

Напишите законченную программу, в которой с помощью подходящего конструктора создайте множество с дубликатами и инициализируйте его четырьмя строковыми значениями. С помощью итераторов выведите содержимое созданного множества на экран. Добавьте в множество еще два строковых значения, одно из которых уже имеется в множестве. Еще раз выведите содержимое множества на экран.

Используйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод, классы *string, set*).

Вариант 4. Ассоциативные контейнеры. Словари (*map)*.

Напишите законченную программу, в которой создайте пустой словарь – телефонную книгу для хранения записей в лексикографическом порядке. Заполните телефонную книгу данными из файла *map.dat*. Каждая строка файла хранит фамилию абонента и телефонный номер-число, разделенные пробелом. Выведите на экран содержимое телефонной книги. Дополните телефонную книгу новой записью и измените один из номеров телефонной книги. Снова выведите на экран содержимое телефонной книги. Выполните поиск в словаре существующего и несуществующего элемента и выведите результат поиска на экран.

Используйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод, классы *string, map*).

Вариант 5. Ассоциативные контейнеры. Словари (*map)*.

Напишите законченную программу, в которой создайте две пустых телефонных книги для хранения записей в лексикографическом порядке. Заполните телефонные книги данными из файла *map1.dat* и *map2.dat*. Каждая строка файла хранит фамилию абонента и телефонный номер-число, разделенные пробелом. Выведите на экран содержимое телефонных книг. Выполните обмен словарей и выведите их на экран. Очистите словари.

Используйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод, классы *string, map*).

Вариант 6. Ассоциативные контейнеры. Словари с дубликатами (*multimap)*.

Напишите законченную программу, в которой создайте пустую телефонную книгу – словарь с дубликатами – для хранения записей в лексикографическом порядке (фамилия абонента – ключ, число – значение, номер телефона). С помощью метода *insert( )* заполните телефонную книгу данными так, чтобы телефонная книга содержала дубликаты с одинаковыми номерами. Выведите на экран содержимое телефонной книги. Выполните поиск в словаре информации с заданными ключами и выведите результаты поиска на экран (ключ с дубликатами, без дубликатов и отсутствующий ключ).

Используйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод, классы *string, map*).

Вариант 7. Ассоциативные контейнеры. Множества (*set)*.

Напишите законченную программу, в которой с помощью подходящего конструктора создайте множество и инициализируйте его четырьмя строковыми значениями. С помощью итераторов выведите содержимое созданного множества на экран. Добавьте в множество еще два строковых значения, одно из которых уже имеется в множестве. Еще раз выведите содержимое множества на экран.

Используйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод, классы *string, set*).

Вариант 8. Ассоциативные контейнеры. Множества с дубликатами (*multiset)*.

Напишите законченную программу, в которой с помощью подходящего конструктора создайте множество с дубликатами и инициализируйте его четырьмя строковыми значениями. С помощью итераторов выведите содержимое созданного множества на экран. Добавьте в множество еще два строковых значения, одно из которых уже имеется в множестве. Еще раз выведите содержимое множества на экран.

Используйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод, классы *string, set*).

Вариант 9. Ассоциативные контейнеры. Словари (*map)*.

Напишите законченную программу, в которой создайте пустой словарь – телефонную книгу для хранения записей в лексикографическом порядке. Заполните телефонную книгу данными из файла *map.dat*. Каждая строка файла хранит фамилию абонента и телефонный номер-число, разделенные пробелом. Выведите на экран содержимое телефонной книги. Дополните телефонную книгу новой записью и измените один из номеров телефонной книги. Снова выведите на экран содержимое телефонной книги. Выполните поиск в словаре существующего и несуществующего элемента и выведите результат поиска на экран.

Используйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод, классы *string, map*).

Вариант 10. Ассоциативные контейнеры. Словари (*map)*.

Напишите законченную программу, в которой создайте две пустых телефонных книги для хранения записей в лексикографическом порядке. Заполните телефонные книги данными из файла *map1.dat* и *map2.dat*. Каждая строка файла хранит фамилию абонента и телефонный номер-число, разделенные пробелом. Выведите на экран содержимое телефонных книг. Выполните обмен словарей и выведите их на экран. Очистите словари.

Используйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод, классы *string, map*).

4.2 Алгоритмы стандартной библиотеки шаблонов

4.2.1 Теоретическое введение

Стандартная библиотека шаблонов предоставляет алгоритмы для выполнения операций, которые требуются пользователю контейнера. Общее число алгоритмов около шестидесяти. Для доступа к алгоритмам библиотеки стандартных шаблонов в программу необходимо включить заголовочный файл <algorithm>.

Алгоритмы стандартной библиотеки шаблонов подразделяются на пять категорий: немодифицированные операции с последовательностями, модифицированные операции с последовательностями, алгоритмы сортировки последовательностей, алгоритмы работы с множествами и пирамидами, обобщенные численные алгоритмы.

Обобщенные численные алгоритмы выполняют обработку числовых элементов. Они помещены в заголовочный файл <numeric> [8, с. 451].

4.2.2 Лабораторная работа №11 «Разработка приложений с использованием обобщенных алгоритмов»

Разработайте программу на языке C++, используя стандартную библиотеку шаблонов.

Вариант 1. Напишите законченную программу, в которой создайте и инициализируйте два целочисленных вектора размерами соответственно 4 и 2 элемента. С помощью метода for_search() и функции show() выведите значения элементов на экран. Функция show() приведена в листинге 1.

Листинг 1. Функция show()

```
// Функция, вызываемая для каждого элемента вектора
void show(int a
{ cout << a << " "; return; }
```

Используйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод, класс *vector,* объявления стандартных функциональных объектов из файла <*functional*> и объявления стандартных алгоритмов из файла <*algorithm*>).

Вариант 2. Напишите законченную программу, в которой создайте и инициализируйте целочисленный вектор. Выведите значения его элементов на экран. С помощью метода *find( )* выполните поиск в векторе элемента с заданным значением и выведите результаты поиска на экран. С помощью метода count( ) подсчитайте в векторе количество элементов с заданным значением и выведите результаты поиска на экран.

Используйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод, класс *vector,* объявления стандартных

функциональных объектов из файла <*functional*> и объявления стандартных алгоритмов из файла <*algorithm*>).

Вариант 3. Напишите законченную программу, в которой создайте и инициализируйте целочисленный вектор. Выведите значения его элементов на экран. С помощью метода *find-if( )* и функции odd() выполните поиск в векторе первого элемента с нечетным значением и выведите результаты поиска на экран. Функция odd() приведена в листинге 2.

Листинг 2. Функция odd()

```
//Функция для поиска первого нечетного элемента вектора
bool odd(// Возвращает true, если нечетное
int a) { if( a%2 )   return true;    return false;}
```

Используйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод, класс *vector,* объявления стандартных функциональных объектов из файла <*functional*> и объявления стандартных алгоритмов из файла <*algorithm*>).

Вариант 4. Напишите законченную программу, в которой создайте и инициализируйте целочисленный вектор. Выведите значения его элементов на экран. С помощью метода *adjacent_find-if( )* выполните поиск в векторе первоой пары элементов со значениями «меньше-больше» и выведите результаты поиска на экран.

Используйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод, класс *vector,* объявления стандартных функциональных объектов из файла <*functional*> и объявления стандартных алгоритмов из файла <*algorithm*>).

Вариант 5. Напишите законченную программу, в которой создайте и инициализируйте два целочисленных вектора *v, v1* размерами соответственно 4 и 2 элемента. Выведите значения элементов на экран. С помощью метода *find-end( )* выполните поиск последнего вхождения *v1* в *v* и выведите результаты поиска на экран, в том числе напечатайте значение найденного элемента. С помощью метода *find-first_of( )* выполните поиск первого вхождения в *v* элемента из *v1* и выведите результаты поиска на экран.

Используйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод, класс *vector,* объявления стандартных функциональных объектов из файла <*functional*> и объявления стандартных алгоритмов из файла <*algorithm*>).

Вариант 6. Напишите законченную программу, в которой создайте и инициализируйте два целочисленных вектора *v, v1* размерами соответственно 4 и 2 элемента. Выведите значения элементов на экран. Проверьте *v, v1* на несовпадение и совпадение с помощью методов *mismatch( )* и *equal( )* соответственно, выведите результаты проверки на экран. С помощью метода *search( )* выполните проверку вхождения в *v1* в *v* и выведите результаты проверки на экран.

Используйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод, класс *vector,* объявления стандартных

функциональных объектов из файла <*functional*> и объявления стандартных алгоритмов из файла <*algorithm*>).

Вариант 7. Напишите законченную программу, в которой создайте два целочисленных массива *a[5] и b[4]*. Первый массив инициализируйте значениями 1, 2, 3, 4, 5. С помощью алгоритма *copy( )* скопируйте последние четыре элемента массива *a* в массив *b*. Выведите значения элементов *b* на экран. С помощью алгоритма *copy( )* скопируйте последние четыре элемента массива *a* в его начало и выведите значения элементов *a* на экран. С помощью алгоритма *copy_backward( )* скопируйте первые три элемента массива *b* справа-налево в этот же массив, начиная копирование в четвертый элемент этого же массива. Выведите значения элементов *b* на экран. С помощью алгоритма *copy( )* выведите на экран значения элементов массива *a.* С этой целью в качестве третьего аргумента в вызове алгоритма *copy( )* используйте потоковый итератор *ostream_iterator< int >(cout, " ")*.

Применяйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод и объявления стандартных алгоритмов из файла <*algorithm*>).

Вариант 8. Напишите законченную программу, в которой создайте целочисленный массив *a[5]*. С помощью алгоритма *fill( )* заполните массив единичными значениями и выведите значения элементов массива на экран. С помощью алгоритма *fill_n( )* заполните третий и четвертый элементы массива нулевыми значениями и выведите значения элементов массива на экран.

Применяйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод и объявления стандартных алгоритмов из файла <*algorithm*>).

Вариант 9. Напишите законченную программу, в которой создайте целочисленный массив *a[5]*. С помощью алгоритма *generate( )* заполните массив значениями, возвращаемыми функцией f() и выведите значения элементов массива на экран. Функция f() приведена в листинге 3.

Листинг 3. Функция f()

```
//Функция, вычисляющая значения для заполнения последовательности
int f( void )
{ static int i = 1; return ( ++i ) * 3; }
```

Применяйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод и объявления стандартных алгоритмов из файла <*algorithm*>).

Вариант 10. Напишите законченную программу, в которой создайте целочисленные массивы *a[5]={1,2,3,4.5}* и *b[5]={11,12,13,14,15}*. С помощью алгоритма *iter_swap( )* поменяйте местами первый элемент *a* и пятый элемент *b* и выведите значения элементов массивов на экран. С помощью этого же алгоритма поменяйте мастами первый и последний элементы массива *a* и выведите значения элементов массива *a* на экран. С помощью алгоритма *swap_ranges( )* поменяйте местами первые два элемента массива *a* с четвертым и пятым элементами массива *b* и выведите значения элементов массивов на

экран. С помощью этого же алгоритма поменяйте местами первые два элемента массива *a* с его четвертым и пятым элеметами и выведите значения элементов массива *a* на экран. Применяйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод и объявления стандартных алгоритмов из файла <*algorithm*>).

Методические указания

1 Создайте проект «Windows Forms Application Visual C++».
2 Разработайте диаграмму классов.
3 Формализуйте алгоритм решения задачи на ПЭВМ.
4 Выведите на экран видеомонитора результаты работы программы.
5 Оформите отчет по лабораторной работе.

4.2.3 Практическая работа №8 «Обобщенные алгоритмы»

Вариант 1. Алгоритмы. Модифицирующие операции с последовательностями. Напишите законченную программу, в которой с помощью подходящих конструкторов создайте два целочисленных вектора *v1* – пустой и *v1* размером 10 элементов с нулевыми значениями. Занесите в вектор *v1* последовательность значений 10, 20, 30, 40, 50, 60, 70, 80, 90. Выведите значения элементов созданных векторов на экран. С помощью алгоритма *replace_copy_if( )* скопируйте *v1* в *v2* с заменой в копии значений больших 30 и меньших 70 на значение 5. С этой целью в качестве третьего аргумента в вызове алгоритма используйте функцию-предикат вида

```
// Функция-предикат для выбора элементов вектора
bool In_30_70(// Возвращает true при 30 < x < 50
int x) //Проверяемое значение
{return x>30 && x<70;}
```

Выведите значения элементов обоих векторов на экран.

Применяйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод, класс *vector* и объявления стандартных алгоритмов из файла <*algorithm*>).

Вариант 2. Алгоритмы. Модифицирующие операции с последовательностями. Напишите законченную программу, в которой создайте два целочисленных массива *a* и *b* одинакового размера и инициализируйте их. Выведите значения элементов созданных массивов на экран. С помощью алгоритма *transform()* выполните попарную обработку элементов этих массивов в соответствии с выражением $a_i = a_i^2 - b_i^2$. С этой целью в качестве пятого аргумента в вызове алгоритма используйте функциональный объект вида

```
//Функциональный объект пользователя, выполняющий
//обработку пар элементов двух последовательностей
struct conv: binary_function <double, double, double>
{double operator() ( double x, double y) const
```

*{return x*x - y*y;}};*

Выведите значения элементов массива *a* на экран. С помощью алгоритма *transform( )* выполните инвертирование значений элементов вектора *b*. С этой целью в качестве четвертого аргумента в вызове алгоритма используйте предикат *negate< int >*. Выведите значения элементов массива *b* на экран.

Применяйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод, объявления стандартных алгоритмов из файла <*algorithm*> и объявления стандартных функциональных объектов из файла <*functional*>).

Вариант 3. Алгоритмы. Модифицирующие операции с последовательностями.

Напишите законченную программу, в которой создайте три целочисленных массива *a, b* и *c* одинакового размера. Инициализируйте массивы *a* и *c* так, чтобы они содержали одинаковые элементы. Выведите значения элементов массивов *a* и *c* на экран. С помощью алгоритма *unique( )* удалите в последовательностях элементов массива *a* повторяющиеся элементы и выведите на экран значения элементов этого массива. С помощью алгоритма *unique_copy( )* выполните копирование массива *c* в массив *b* с удалением в последовательностях элементов массива *b* повторяющихся элементов и выведите на экран значения элементов массива *b*.

Применяйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод и объявления стандартных алгоритмов из файла <*algorithm*>).

Вариант 4. Алгоритмы, связанные с сортировкой.

Напишите законченную программу, в которой с помощью подходящих конструкторов создайте два вектора вещественного типа *v, v1* и *v2*. Инициализируйте первые два вектора случайными значениями, а вектор *v2* так, чтобы он содержал две отсортированных последовательности. Выведите значения элементов этих векторов на экран. С помощью алгоритмов *sort()* и *stable_sort()* выполните соответственно обычную и устойчивую сортировку векторов *v, v1* и выведите на экран значения элементов этих векторов. С помощью алгоритма *binary_search( )* выполните поиск в векторе *v* заданного значения и выведите результаты поиска на экран. С помощью алгоритма *implace_merge( )* выполните слияниие двух отсортированных последовательностей в векторе *v2* и выведите значения элементов этого вектора экран. Применяйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод и объявления стандартных алгоритмов из файла <*algorithm*>).

Вариант 5. Алгоритмы, связанные с сортировкой.

Напишите законченную программу, в которой с помощью подходящих конструкторов создайте два вектора вещественного типа *v, v1* одинакового размера. Инициализируйте оба вектора отсортированными и одинаковыми значениями. Выведите значения элементов этих векторов на экран. С помощью алгоритма *lexicogpaphical_compare( )* сравните вектора *v, v1* между собой и

выведите на экран результаты сравнения. Измените значение одного из элементов вектора *v1*. С помощью того же алгоритма сравните вектора *v, v1* между собой с использованием предиката *greater<...>* и выведите на экран результаты сравнения. С помощью алгоритмов *lower_bound( )* и *upper_bound( )* определите соответственно значения элементов вектора *v*, после которого и перед которым можно вставить элемент с заданным значением, не нарушая его упорядоченности, и выведите значения элементов вектора *v* на экран.

Применяйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод, объявления стандартных алгоритмов из файла <*algorithm*> и объявления стандартных функциональных объектов из файла <*functional*>).

Вариант 6. Алгоритмы, связанные с сортировкой.

Напишите законченную программу, в которой с помощью подходящих конструкторов создайте два вектора вещественного типа *v1, v2* размером 3. Инициализируйте оба вектора так, чтобы они были отсортированными. С помощью подходящего конструктора создайте вещественный вектор *v3* размером 6. Выведите значения элементов векторов *v1, v2* на экран. С помощью алгоритмов *max_element( )* и *min_element( )* определите и выведите на экран соответственно максимальное и минимальное значения элементов вектора *v1*. С помощью алгоритма *merge( )* слейте отсортированные векторы *v1, v2* в вектор *v3* и выведите на экран значения элементов вектора *v3*. С помощью алгоритмов *next_permutation( )* и *prev_permutation( )* сгенерируйте и выведите на экран перестановки элементов в лексикографическом порядке для векторов *v1, v2* соответственно.

Применяйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод и объявления стандартных алгоритмов из файла <*algorithm*>).

Вариант 7. Алгоритмы работы с множествами.

Напишите законченную программу, в которой создайте целочисленные массив *a* со значениями элементов 2, 5, 7, 9, массив *b* со значениями элементов 1, 5, 9 и массив *tmp* из семи элементов. Выведите значения элементов массивов *a, b* на экран. С помощью алгоритма *set_intersection( )* создайте отсортированное пересечение массивов *a, b,* поместите его в массив *tmp* и выведите на экран значения элементов массива *tmp*. С помощью алгоритма *set_union( )* создайте отсортированное объединение массивов *a, b,* поместите его в массив *tmp* и выведите на экран значения элементов массива *tmp*. С помощью алгоритма *set_difference( )* скопируйте в *tmp* из массивов *a, b* значения элементов, входящих только в массив *a*, и выведите на экран значения элементов массива *tmp*. С помощью алгоритма *sym_difference( )* скопируйте в *tmp* из массивов *a, b* значения элементов, входящих только в один из массивов, и выведите на экран значения элементов массива *tmp*. С помощью алгоритма *includes( )* проверьте включение массива *b* в массив *a* и выведите на экран результаты проверки.

Применяйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод и объявления стандартных алгоритмов из файла <*algorithm*>).

Вариант 8. Алгоритмы работы с пирамидами.

Напишите законченную программу, в которой с помощью подходящего конструктора создайте целочисленные вектор *Number* со значениями элементов 4, 10, 70, 10, 30, 69, 96, 100 и выведите значения элементов вектора на экран. С помощью алгоритма *make_heap( )* преобразуйте *Number* в пирамиду и выведите на экран значения элементов *Number*. С помощью алгоритма *sort_heap( )* преобразуйте пирамиду в отсортированную последовательность и выведите на экран значения элементов *Number*. С помощью метода *push_back( )* и алгоритма *push_heap( )* добавьте элемент в *Number* и выведите на экран значения элементов *Number*. С помощью алгоритма *make_heap( )* преобразуйте *Number* в пирамиду и выведите на экран значения элементов *Number*. Извлеките и выведите на экран корневой элемент пирамиды. Затем с помощью алгоритма *pop_heap( )* восстановите пирамиду и выведите на экран значения элементов *Number*.

Применяйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод, класс *vector* и объявления стандартных алгоритмов из файла <*algorithm*>).

Вариант 9. Алгоритмы. Модифицирующие операции с последовательностями. Напишите законченную программу, в которой создайте три целочисленных массива *a, b* и *c* одинакового размера. Инициализируйте массивы *a* и *c* так, чтобы они содержали одинаковые элементы. Выведите значения элементов массивов *a* и *c* на экран. С помощью алгоритма *unique( )* удалите в последовательностях элементов массива *a* повторяющиеся элементы и выведите на экран значения элементов этого массива. С помощью алгоритма *unique_copy( )* выполните копирование массива *c* в массив *b* с удалением в последовательностях элементов массива *b* повторяющихся элементов и выведите на экран значения элементов массива *b*.

Применяйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод и объявления стандартных алгоритмов из файла <*algorithm*>).

Вариант 10. Алгоритмы, связанные с сортировкой.

Напишите законченную программу, в которой с помощью подходящих конструкторов создайте два вектора вещественного типа *v, v1* и *v2*. Инициализируйте первые два вектора случайными значениями, а вектор *v2* так, чтобы он содержал две отсортированных последовательности. Выведите значения элементов этих векторов на экран. С помощью алгоритмов *sort()* и *stable_sort()* выполните соответственно обычную и устойчивую сортировку векторов *v, v1* и выведите на экран значения элементов этих векторов. С помощью алгоритма *binary_search( )* выполните поиск в векторе *v* заданного значения и выведите результаты поиска на экран. С помощью алгоритма *implace_merge()* выполните слияние двух отсортированных

последовательностей в векторе *v2* и выведите значения элементов этого вектора на экран.

Применяйте только средства стандартной библиотеки языка C++ (потоковый ввод-вывод и объявления стандартных алгоритмов из файла <*algorithm*>).

ЗАКЛЮЧЕНИЕ

При создании программ применяются различные технологии программирования: структурная, объектно-ориентированная, обобщенная с использованием стандартной библиотеки.

Технология визуального проектирования и событийного программирования позволяет создавать эффективные приложения, с которыми связаны два аспекта: программный код для создания графического интерфейса пользователя, с которым взаимодействует пользователь, и программный код для обработки этого взаимодействия и реализации полезной функциональности приложения.

Методические указания к выполнению лабораторных и практических работ позволяют студентам закрепить теоретический материал по дисциплине «Основы программирования» и приобрести практические навыки в разработке визуальных приложений на языке C++.

СПИСОК ЛИТЕРАТУРЫ

1 Хортон А. Visual C++ 2010. Полный курс / пер. с англ. В. Коваленко. – М. : ООО «И. Д. Вильямс», 2011. – 1216 с.

2. Зиборов В. В. MS Visual C++ 2010 в среде .NET. Библиотека программиста. – СПб. : Питер, 2012. – 320 с.

3 Культин Н. Б. Основы программирования в Microsoft Visual C++ 2010. – СПб. : БХВ-Петербург, 2010. – 384 с.

4 Культин Н. Б. Microsoft Visual C++ в задачах и примерах. – СПб. : БХВ-Петербург, 2010. – 272 с.

5. Пахомов Б. И. C/C++ и MS Visual C++ 2012 для начинающих. – СПб. : БХВ-Петербург, 2013 – 512 с.

6. Гагарина Л. Г., Колдаев В. Д. Алгоритмы и структуры данных : учебное пособие. – М. : Финансы и статистика, Инфра-м, 2009. – 304 с.

7 Анашкина Н. В. Технологии и методы программирования. – М. : Издательский центр «Академия», 2012. – 384 с.

8 Давыдов В. Г. Технологии программирования. C++. – СПб. : БХВ-Петербург, 2005. – 672 с.

Семахин Андрей Михайлович

ОСНОВЫ ПРОГРАММИРОВАНИЯ

Методические указания
к выполнению лабораторных и практических работ
для студентов направления подготовки 09.03.04
«Программная инженерия»,
10.05.03 «Информационная безопасность автоматизированных систем»
Часть 2

Редактор О. Г. Арефьева

Подписано в печать	Формат 60x84 1/16	Бумага 65 г/м^2
Печать цифровая	Усл. печ. л. 3,0	Уч.-изд. л. 3,0
Заказ	Тираж 25	Не для продажи

РИЦ Курганского государственного университета.
640000, г. Курган, ул. Советская, 63/4.
Курганский государственный университет.