
# КОНСТРУИРОВАНИЕ ПРОГРАММ И ЯЗЫКИ ПРОГРАММИРОВАНИЯ

**УЧЕБНАЯ ПРОГРАММА, МЕТОДИЧЕСКИЕ УКАЗАНИЯ И КОНТРОЛЬНЫЕ ЗАДАНИЯ**



**МИНСК 2007**

МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ
«МИНСКИЙ ГОСУДАРСТВЕННЫЙ ВЫСШИЙ
РАДИОТЕХНИЧЕСКИЙ КОЛЛЕДЖ»

ПОДЛЕЖИТ ВОЗВРАТУ

УТВЕРЖДАЮ
Проректор по учебной работе
_____ В. И. Федосенко
« 19 » апреля 2007 г.

# КОНСТРУИРОВАНИЕ ПРОГРАММ И ЯЗЫКИ ПРОГРАММИРОВАНИЯ

Учебная программа, методические указания
и контрольные задания
для студентов безотрывной формы обучения
специальности 1-08 01 01-07
«Профессиональное обучение. (Информатика)»



МИНСК 2007

УДК 621.3.06(075)
ББК 32.973.26–018я7
К65

Рекомендовано к изданию кафедрой информатики и Научно-методическим советом Учреждения образования «Минский государственный высший радиотехнический колледж»

С о с т а в и т е л и:
**М. А. Бельчик**, ассистент кафедры информатики МГВРК,
**В. А. Заневская**, старший преподаватель кафедры информатики

Р е ц е н з е н т
**Ю. А. Скудняков**, заведующий кафедрой информатики МГВРК, канд. техн. наук, доцент

**Конструирование** программ и языки программирования : учеб. программа, метод. указания и контрол. задания для студентов безотрыв. формы обучения специальности 1-08 01 01-07 «Профессиональное обучение. (Информатика)» / сост. М. А. Бельчик, В. А. Заневская. – Мн. : МГВРК, 2007. – 52 с.
ISBN 978-985-6754-91-6

Рассмотрена программа дисциплины, даны вопросы для самоподготовки и варианты контрольной работы, приведены методические указания по выполнению и оформлению контрольной работы, основные теоретические сведения по выполнению практических заданий, рекомендуемая литература.

Предназначено для студентов и преподавателей колледжа.

**УДК 621.3.06(075)**
**ББК 32.973.26–018я7**



**ISBN 978-985-6754-91-6**

**Предисловие**

Учебная дисциплина «Конструирование программ и языки программирования» является одной из основных в цикле специальных и базируется на знаниях и навыках, полученных при изучении следующих дисциплин: «Информатика», «Операционные системы», «Основы алгоритмизации и программирования». Реализация заданий с помощью ЭВМ, осуществляемая на практических занятиях, опирается на знание методов и средств программирования, умения разрабатывать алгоритмы.

Программа дисциплины «Конструирование программ и языки программирования» ставит своей целью выработку профессиональных навыков разработки и программирования задач различного уровня с использованием современных методов и средств.

Рабочая программа состоит из 4-х основных разделов:

- язык программирования Ассемблер;
- профессиональное программирование на C/C++;
- объектный подход к разработке программ;
- методы компоновки программных модулей.

Рассмотрим каждый более подробно.

РАЗДЕЛ 1. **ЯЗЫК ПРОГРАММИРОВАНИЯ АССЕМБЛЕР**

*Цели обучения*:

- дать представление об архитектуре машинно-ориентированных языков;
- научить применять ассемблерные коды в программах высокого уровня.

В результате изучения материала *студент должен*:

- иметь представление о машинных кодах, системе команд, организации и методах доступа к памяти ПК;
- разбираться и уметь объяснить принцип действия программы, написанной на Ассемблере;
- составить, отладить и выполнить программу по обработке действий;
- применять при необходимости ассемблерные вставки в программах на языках С, Паскаль и т. д.

РАЗДЕЛ 2. **ПРОФЕССИОНАЛЬНОЕ ПРОГРАММИРОВАНИЕ НА C/C++**

*Цели обучения*:

- сформировать правильный стиль программирования;

- выработать творческий подход к решению задач.

В результате изучения материала *студент должен*:

- знать и уметь применять базовые конструкции языка при решении типовых задач;

- иметь представление о структурировании программ и данных;

- применять умения и навыки в новой ситуации.

РАЗДЕЛ 3. **ОБЪЕКТНЫЙ ПОДХОД К РАЗРАБОТКЕ ПРОГРАММ**

*Цели обучения*:

- дать представление об объектном подходе к проектированию и программированию;

- сформировать навыки разработки объектных программ.

В результате изучения материала *студент должен*:

- уметь провести объектный анализ проблемы (задачи);

- выразить в виде определенных классов абстракции предметной области;

- применять объектные возможности языка для реализации программ с использованием классов.

РАЗДЕЛ 4. **МЕТОДЫ КОМПОНОВКИ ПРОГРАММНЫХ МОДУЛЕЙ**

*Цели обучения*:

- дать представление о методах компоновки разноязыковых программ;

- научить применять готовые библиотечные модули для решения конкретных задач пользователя.

В результате изучения материала *студент должен*:

- иметь представление об особенностях организации межмодульных связей;

- знать способы передачи параметров, характерных для языков С и Паскаль;

- уметь использовать возможности библиотек в своих программах.

По результатам изучения дисциплины *студенты должны*:

- проводить структурный анализ задачи;

- проводить объектный анализ задачи;

- обосновывать выбранные пути реализации задачи;

- использовать возможности различных библиотек;

- разрабатывать надежные, работоспособные программы;

- использовать автоматизированные средства при разработке программ.

Завершающим этапом изучения дисциплины является выполнение курсового проекта.

Учебный план специальности «Профессиональное обучение. (Информатика)» предусматривает изучение дисциплины «Конструирование программ и языки программирования» в объеме 149 часов, из них 80 часов – лабораторные занятия.

# 1. Учебная программа

## 1.1. Тематический план дисциплины

Таблица 1

| Наименование раздела и темы | Всего часов | | | |
|---|---|---|---|---|
| | дневная форма обучения | | безотрывная форма обучения | |
| | теория | практика | теория | практика |
| Введение | – | – | – | – |
| РАЗДЕЛ 1. **ЯЗЫК ПРОГРАММИРОВАНИЯ АССЕМБЛЕР** | 26 | 18 | – | 2 |
| 1.1. Архитектура и система команд процессоров | 18 | – | – | – |
| 1.2. Базовые директивы | 2 | – | – | – |
| 1.3. Система прерываний и программирование ввода-вывода | 2 | – | – | – |
| 1.4. Программирование с использованием математического сопроцессора | 4 | – | – | – |
| РАЗДЕЛ 2. **ПРОФЕССИОНАЛЬНОЕ ПРОГРАММИРОВАНИЕ НА C/C++** | 23 | 20 | 8 | 8 |
| 2.1. История развития языка | 1 | – | – | – |
| 2.2. Лексемы | 2 | – | – | – |
| 2.3. Ввод-вывод | 4 | – | 2 | – |
| 2.4. Синтаксические операторы | 2 | – | 1 | – |
| 2.5. Функции | 2 | – | 2 | – |
| 2.6. Структурные типы данных | 6 | – | 2 | – |
| 2.7. Указатели | 4 | – | 1 | – |
| 2.8. Обращение к ресурсам системы | 2 | – | – | – |

Окончание табл. 1

| Наименование раздела и темы | Всего часов | | | |
|---|---|---|---|---|
| | дневная форма обучения | | безотрывная форма обучения | |
| | теория | практика | теория | практика |
| РАЗДЕЛ 3. **ОБЪЕКТНЫЙ ПОДХОД К РАЗРАБОТКЕ ПРОГРАММ** | 14 | 34 | 6 | 8 |
| 3.1. Введение в классы | 2 | – | 1 | – |
| 3.2. Конструктор. Деструктор | 2 | – | 1 | – |
| 3.3. Специальный полиморфизм | 2 | – | 1 | – |
| 3.4. Наследование | 6 | – | 2 | – |
| 3.5. Абстрактный тип данных | 2 | – | 1 | – |
| РАЗДЕЛ 4. **МЕТОДЫ КОМПОНОВКИ ПРОГРАММНЫХ МОДУЛЕЙ** | 6 | 8 | – | – |
| 4.1. Компоновка при структурном подходе. Проект программы | 2 | – | – | – |
| 4.2. Включение классов из готовых библиотек | 2 | – | – | – |
| 4.3. Смешанное программирование | 2 | – | – | – |
| *Курсовой проект* | 30 | | – | |
| ИТОГО | 69 | 80 | 14 | 18 |
| ВСЕГО | 179 | | 32 | |

## 1.2. Содержание дисциплины

**Введение**

Содержание и задачи дисциплины. Связь с другими дисциплинами. – 1 час.

РАЗДЕЛ 1. **ЯЗЫК ПРОГРАММИРОВАНИЯ АССЕМБЛЕР**

ТЕМА 1.1. **АРХИТЕКТУРА И СИСТЕМА КОМАНД ПРОЦЕССОРОВ**

Архитектура ЭВМ. Набор регистров. – 2 часа.

Литература [2, с. 45–56]; [7, с. 26–46].

Оперативная память и операционные системы. – 2 часа.

Литература [2, с. 56–61]; [7, с. 46–51].

Жизненный цикл программы на Ассемблере. Трансляция программы. Компоновка программы. Отладка программы. Утилита MAKE. – 2 часа.

Литература [7, с. 66–86].

Структура машинной команды. Способы задания операндов команды. Функциональная классификация машинных команд. – 2 часа.

Литература [7, с. 123–133].

Команды пересылок. Работа с адресами и указателями. Преобразование данных. Работа со стеком. – 2 часа.

Литература [2, с. 70–81]; [7, с. 133–147].

Арифметические команды. Сложение двоичных чисел со знаком и без знака. Вычитание двоичных чисел со знаком и без знака. Умножение двоичных чисел со знаком и без знака. Деление двоичных чисел со знаком и без знака. Неупакованные и упакованные BCD-числа. – 2 часа.

Литература [2, с. 81–103]; [7, с. 159–180].

Команды обработки строк. – 1 час.

Литература [2, с. 223–233]; [7, с. 229–247].

Команды организации переходов. – 2 часа.

Литература [2, с. 183–200]; [7, с. 202–208, 217–222].

Команды организации циклов.

Литература [2, с. 200–211]; [7, с. 222–229].

Процедуры – 2 часа.

Литература [7, с. 212–217].

### Тема 1.2. **Базовые директивы**

Разница между директивами и командами Ассемблера. Директива Segment. Директива группирования сегментов Group. Директива Assume. Директива Model. Директива Public. Директива Extern. – 2 часа.

Литература [2, с. 61–70]; [7, с. 98–105].

### Тема 1.3. **Система прерываний и программирование ввода-вывода**

Понятие «система прерываний». Вектор прерывания. Программные прерывания. Аппаратные прерывания. Механизм реализации процессов ввода-вывода. Способы доступа к внешним устройствам. Ввод-вывод через порты in, out. Ввод-вывод через прерывания. Резидентная программа и ее особенности. – 2 часа.

Литература [2, с. 312–357]; [7, с. 353–380].

ТЕМА 1.4. **ПРОГРАММИРОВАНИЕ С ИСПОЛЬЗОВАНИЕМ МАТЕМАТИЧЕСКОГО СОПРОЦЕССОРА**

Форматы данных. Система команд сопроцессора. – 2 часа.

Литература [7, с. 509–549, с. 266–285].

Исключения сопроцессора и их обработка. Использование отладчика. – 2 часа.

Литература [2, с. 285–312]; [7, с. 554–561].

## РАЗДЕЛ 2. ПРОФЕССИОНАЛЬНОЕ ПРОГРАММИРОВАНИЕ НА C/C++

ТЕМА 2.1. **ИСТОРИЯ РАЗВИТИЯ ЯЗЫКА**

История развития языка, предшественники, достоинства, интегрированная среда программирования. – 1 час.

Литература [3, с. 38–46].

ТЕМА 2.2. **ЛЕКСЕМЫ**

Пять разновидностей лексем: ключевые слова, идентификаторы, константы, операторы, знаки пунктуации. Простые типы. Инициализация и объявление переменной. Автоматическое и явное преобразование типов. Тип выражения. Арифметические, логические, выражения отношения. Тернарная операция. Операторы для работы с выражениями. Таблица приоритета и ассоциативности. – 2 часа.

Литература [3, с. 113–117, с. 137–141].

ТЕМА 2.3. **ВВОД-ВЫВОД**

Библиотеки stdio и iostream. Разделители. Манипуляторы. – 2 часа.

Литература [3, с. 629–664]; [6, с. 242–275].

Файловый ввод-вывод. – 2 часа.

Литература [3, с. 737–756]; [6, с. 275–302].

ТЕМА 2.4. **СИНТАКСИЧЕСКИЕ ОПЕРАТОРЫ**

Составной, пустой операторы. Оператор условия if. Многоканальный условный оператор switch. Оператор разрешения видимости «::».

Литература [3, с. 93–95, с. 128–134, с. 221–222, с. 416–417].

Операторы повтора for, while, do/while, передачи управления goto, continue, break. – 2 часа.

Литература [3, с. 99–100, с. 120–124, с. 134–136].

ТЕМА 2.5. **ФУНКЦИИ**

Определение функции. Оператор return. Прототип функции. Аргументы по умолчанию. Тип и класс памяти функций. – 2 часа.

Литература [3, с. 181–187, с. 198–201, с. 219–221].

Математические библиотечные функции. Рекурсия. Рекурсия или итерация – 2 часа.

Литература [3, с. 180–181, с. 211–213].

ТЕМА 2.6. **СТРУКТУРНЫЕ ТИПЫ ДАННЫХ**

Массивы. – 2 часа.

Литература [3, с. 259–290]; [5, с. 79–82].

Строки. Обработка строк. – 2 часа.

Литература [3, с. 358–367]; [5, с. 82–88].

Структуры. Объединения. – 2 часа.

Литература [3, с. 406–409]; [5, с. 88–95].

ТЕМА 2.7. **УКАЗАТЕЛИ**

Указатели. Адресация и разыменование. Объявление ссылок и вызов по ссылке. Массивы и указатели. Указатели и структуры. – 2 часа.

Литература [3, с. 322–348]; [5, с. 95–99].

Арифметика указателей. Указатели и функции. – 2 часа.

Литература [3, с. 340–344, с. 354–358]; [5, с. 103–106].

ТЕМА 2.8. **ОБРАЩЕНИЕ К РЕСУРСАМ СИСТЕМЫ**

Ближние и дальние указатели. Операторы управления памятью new, delete. Нормализованный адрес. Указатели на системные области. Функции управления памятью. – 2 часа.

Литература [3, с. 479–480]; [5, с. 99–103].

## РАЗДЕЛ 3. ОБЪЕКТНЫЙ ПОДХОД К РАЗРАБОТКЕ ПРОГРАММ

ТЕМА 3.1. **ВВЕДЕНИЕ В КЛАССЫ**

Объявление класса. Методы доступа к членам класса. Инициализация объекта. Область действия класса и доступ к членам класса. – 2 часа.

Литература [3, с. 411–426]; [5, с. 247–255].

Тема 3.2. **Конструктор. Деструктор**

Специальные методы инициализации объектов. Массивы объектов. Использование конструкторов с аргументами по умолчанию. Использование деструкторов. Вызов конструкторов и деструкторов. – 2 часа.

Литература [3, с. 429–443]; [5, с. 255–268].

Тема 3.3. **Специальный полиморфизм**

Перегрузка и выбор функции. Перегрузка операторов. Соответствие сигнатуре (списку типов параметров функции). – 2 часа.

Литература [3, с. 498–531]; [5, с. 275–280]; [6, с. 175–197].

Тема 3.4. **Наследование**

Базовые классы и производные классы. Защищенные члены класса. Приведение типов указателей базовых классов к указателям производных классов. – 2 часа.

Литература [3, с. 554–562].

Использование функций-членов. Переопределение членов базового класса в производном классе. Прямые и косвенные базовые классы. – 2 часа.

Литература [3, с. 562–568].

Использование конструкторов и деструкторов в производных классах. Неявное преобразование объектов производных классов в объекты базовых классов. Множественное наследование.

Литература [3, с. 568–573, с. 580–586].

Тема 3.5. **Абстрактный тип данных**

Дружественные функции. Виртуальные функции. – 2 часа.

Литература [3, с. 595–614]; [6, с. 97–103].

## Раздел 4. Методы компоновки программных модулей

Тема 4.1. **Компоновка при структурном подходе. Проект программы**

Внешние функции и способы передачи параметров. Организация связей в С. Организация связей в Паскале. Отладка многоязыковых программ. Использование проекта. – 2 часа.

Литература [10, с. 349–351].

ТЕМА 4.2. **ВКЛЮЧЕНИЕ КЛАССОВ ИЗ ГОТОВЫХ БИБЛИОТЕК**

Обзор и возможности использования ресурсов библиотек, ориентированных на C++. – 2 часа.

Литература [9, с. 212–232].

Обзор и возможности использования ресурсов библиотек, ориентированных на приложения Windows. – 2 часа.

Литература [8, с. 210–248].

ТЕМА 4.3. **СМЕШАННОЕ ПРОГРАММИРОВАНИЕ**

Методы компоновки программ при смешивании структурного подхода. – 2 часа.

Литература [8, с. 210–248].

## 1.3. Перечень тем лабораторных занятий

В табл. 2 приведен перечень тем лабораторных занятий.

Таблица 2

| Название работы | Количество часов | |
|---|---|---|
| | дневная форма обучения | безотрывная форма обучения |
| 1. Разработка примеров на Ассемблере | 2 | 2 |
| 2. Разработка простых программ с арифметикой | 2 | – |
| 3. Умножение двоичных 32-битовых чисел без знака | 2 | – |
| 4. Обработка строк | 2 | – |
| 5. Разработка процедур | 2 | – |
| 6. Разработка макросов | 4 | – |
| 7. Реализация TSR-программ | 2 | – |
| 8. Ввод-вывод с использованием различных операторов, манипуляторов, разделителей | 2 | – |
| 9. Выражения. Приведение типов | 2 | – |
| 10. Разработка простейших программ на C++ | 2 | – |
| 11. Реализация функций | 2 | 2 |
| 12. Указатели на функцию | 2 | – |
| 13. Массивы | 2 | – |
| 14. Массивы и указатели | 2 | 2 |

Окончание табл. 2

| Название работы | Количество часов | |
|---|---|---|
| | дневная форма обучения | безотрывная форма обучения |
| 15. Строки | 2 | 2 |
| 16. Структуры | 2 | – |
| 17. Работа с файлами | 2 | 2 |
| 18. Декларирование классов | 2 | 1 |
| 19. Разработка объектно-ориентированных программ на примере арифметических функций | 2 | 1 |
| 20. Разработка объектно-ориентированных программ с перегрузкой функций | 2 | 2 |
| 21. Разработка объектно-ориентированных программ с наследованием | 2 | 2 |
| 22. Разработка объектно-ориентированных программ с виртуальными функциями | 2 | 2 |
| 23. Создание Windows-приложения | 2 | – |
| 24. Компиляция и отладка приложения | 2 | – |
| 25. Окна среды Visual Studio .NET | 2 | – |
| 26. Ресурсы проектов и решений | 2 | – |
| 27. Операции с файлами и получение справки | 2 | – |
| 28. Создание графического редактора | 4 | – |
| 29. Реализация документа | 2 | – |
| 30. Реализация средств ввода-вывода | 2 | – |
| 31. Создание текстового редактора | 2 | – |
| 32. Прокрутка и разделение окон представления | 2 | – |
| 33. Панели инструментов и строка состояния | 2 | – |
| 34. Модальные и немодальные диалоговые окна | 2 | – |
| 35. Компоновка программных модулей | 2 | – |
| 36. Вызов функций Ассемблера из С и Паскаля | 2 | – |
| 37. Разработка прикладных модулей с MFC | 2 | – |
| 38. Включение С, Ассемблера, Паскаля | 2 | – |
| ИТОГО | 80 | 18 |

## 2. Вопросы для самоконтроля

1. Архитектура ЭВМ.
2. Набор регистров. Оперативная память.
3. Операционные системы.
4. Жизненный цикл программы на Ассемблере.
5. Трансляция программы. Компоновка программы. Отладка программы. Утилита MAKE.
6. Структура машинной команды. Способы задания операндов команды. Функциональная классификация машинных команд.
7. Команды пересылок. Работа с адресами и указателями. Преобразование данных. Работа со стеком.
8. Арифметические команды.
9. Команды обработки строк.
10. Команды организации переходов.
11. Команды организации циклов.
12. Базовые директивы. Разница между директивами и командами Ассемблера.
13. Директива Segment.
14. Директива группирования сегментов Group.
15. Директива Assume.
16. Директива Model.
17. Директива Public.
18. Директива Extern.
19. Понятие «система прерываний». Вектор прерывания.
20. Программные прерывания.
21. Аппаратные прерывания.
22. Механизм реализации процессов ввода-вывода.
23. Способы доступа к внешним устройствам.
24. Ввод-вывод через порты in, out. Ввод-вывод через прерывания.
25. Резидентная программа и ее особенности.
26. Программирование с использованием математического сопроцессора.
27. Форматы данных. Система команд сопроцессора.
28. Исключения сопроцессора и их обработка.
29. Использование отладчика.
30. Структурное программирование.
31. Принципы объектно-ориентированного программирования.
32. Структура программы на C++. Составной оператор, блок.

33. История создания и развития языка C++. Алфавит языка C++. Ключевые слова.
34. Типы данных в C++, объявление переменных. Описание констант.
35. Преобразование типов.
36. Основные операции работы с выражениями и их приоритет.
37. Операция присваивания. Составное присваивание.
38. Побитовые и логические операции.
39. Операции отношения, инкремента и декремента.
40. Условная операция. Операция sizeof.
41. Операция разрешения области действия. Операция индексации.
42. Ввод-вывод в языке C. Библиотека stdio.h.
43. Функции символьного ввода-вывода.
44. Ввод-вывод в языке C++. Библиотека iostream.h.
45. Манипуляторы, разделители. Создание пользовательских манипуляторов.
46. Файловый ввод-вывод.
47. Операторы выбора: if, if/else, switch.
48. Операторы повтора: for, while, do/while.
49. Операторы передачи управления: continue, break, goto.
50. Массивы. Описание. Инициализация. Обработка.
51. Строки. Функции работы со строками.
52. Структуры. Описание. Доступ к элементам структуры.
53. Указатели. Операции над указателями.
54. Арифметика указателей. Массивы и указатели.
55. Библиотечные функции и заголовочные файлы.
56. Функции пользователя. Описания. Прототипы.
57. Аргументы по умолчанию. Возвращаемые значения.
58. Перегрузка имени функции.
59. Шаблоны функций.
60. Встраиваемые функции.
61. Способы передачи аргументов в функции.
62. Классы памяти и область действия.
63. Объявление класса. Методы доступа к членам класса.
64. Спецификаторы доступа.
65. Конструкторы и деструкторы.
66. Конструктор копирования.
67. Объект. Инициализация объекта.
68. Указатели. Ссылки и массивы объектов.

69. Объекты в качестве возвращаемых значений функций.
70. Одиночное наследование.
71. Множественное наследование.
72. Спецификаторы наследования.
73. Дружественные функции и классы.
74. Виртуальные функции.
75. Виртуальные базовые классы.
76. Работа с экземплярами классов через указатели, переопределение членов базового класса в производном классе.
77. Перегрузка унарных операторов.
78. Перегрузка бинарных операторов.
79. Обработка исключительных ситуаций.
80. Внешние функции и способы передачи параметров.
81. Организация связей в С.
82. Организация связей в Паскале.
83. Отладка многоязыковых программ.
84. Использование проекта.
85. Возможности использования ресурсов библиотек, ориентированных на С++.
86. Обзор и возможности использования ресурсов библиотек, ориентированных на Windows-приложения.
87. Методы компоновки программ при смешивании структурного подхода.

## 3. Контрольная работа

Учебным планом специальности 1-08 01 01-07 «Профессиональное обучение. (Информатика)» для студентов безотрывной формы обучения предусмотрено выполнение контрольной работы по дисциплине «Конструирование программ и языки программирования».

Главная цель контрольной работы – показать умения студента самостоятельно работать с литературой, проводить анализ, проектирование и программирование задачи.

Выполнение домашней контрольной работы требует от студентов безотрывной формы обучения самостоятельного изучения материала по пособиям, книгам, конспекту.

При необходимости студент может получить консультацию у преподавателя.

*При оформлении работы и выборе заданий следует руководствоваться следующими правилами:*

1. Контрольная работа должна быть выполнена строго в соответствии с предложенным вариантом.

2. Вариант контрольной работы определяется порядковым номером студента в учебном журнале группы.

3. Титульный лист контрольной работы должен содержать фамилию, имя, отчество и номер группы студента, название дисциплины, номер варианта, фамилию преподавателя-рецензента.

4. Перед выполнением каждого задания полностью выписывается его условие, соответствующее варианту.

5. Задания надо располагать в порядке возрастания их номеров.

6. Контрольная работа должна содержать четыре выполненных задания: два теоретических вопроса и две практические задачи. В конце контрольной работы обязательно наличие списка использованной литературы.

7. Теоретическую часть контрольной работы следует выполнять в текстовом редакторе Microsoft Word.

8. Ответы на вопросы должны быть конкретными, четкими, не допускающими двойственного истолкования, сопровождаться примерами, содержать собственные выводы и рассуждения, а также ссылки на использованную литературу с указанием страниц в соответствии с порядковым номером источника в списке литературы (например [6, с. 32–37]).

9. Программирование задач необходимо осуществлять в среде Borland C++ либо в Microsoft Visual C++ с помощью Win32 Console Application.

10. Задачи надо разбивать на подзадачи, которые необходимо реализовать в виде функций в нотации C++. Разработанные программы должны быть правильно структурированы и содержать комментарии.

11. На проверку необходимо представить отчет по контрольной работе на бумажном носителе, а также электронный вариант. Отчет должен быть подготовлен в текстовом редакторе Microsoft Word и содержать четыре выполненных задания: два теоретических вопроса и две практические задачи. В отчете необходимо представить также тексты разработанных программ и результаты их работы в виде копий экранных форм. На маг-

нитном или оптическом носителе в папке (каталоге) (имя папки (каталога) – это фамилия студента, выполнившего контрольную работу) надо представить отчет по контрольной работе, исходные и исполняемые файлы разработанных программ.

Контрольная работа, содержащая четыре верно выполненных задания в соответствии с вариантом и правилами, представленными выше, идет в зачет.

«Не зачтено» выставляется в том случае, если теоретические вопросы раскрыты поверхностно, отсутствуют ссылки на литературу, нет примеров и выводов, практические задания содержат грубые ошибки, а результаты отсутствуют или неверные.

## 4. Методические указания по выполнению заданий контрольной работы

***Задания 1 и 2.*** Требуется исчерпывающе ответить на вопросы. Привести примеры.

***Задание 3.*** Написать программу, которая должна позволять:

- сохранять вводимые данные в текстовом файле;
- просматривать, корректировать, удалять записи из файла;
- выводить результаты работы программы на экран и сохранять в другой текстовый файл.

Программа должна продемонстрировать работу с файлами произвольного доступа.

При необходимости в программу можно добавлять недостающую информацию, но она не должна быть избыточной.

***Задание 4.*** Программа должна содержать:

- базовый класс, включающий два элемента *x* и *y* типа *double*;
- конструктор с параметрами для создания объектов;
- деструктор;
- виртуальные методы просмотра текущего состояния и переустановки объектов базового класса в новое состояние;
- производный класс, включающий элемент *z* типа *double*;
- конструктор с параметрами и списком инициализаторов, передающий данные конструктору базового класса;
- переопределенные методы просмотра текущего состояния

объектов и их переустановки в новое состояние.

Программа должна продемонстрировать работу конструкторов базового и производного классов. Начальное значение задается конструкторами, а переустановка их в новое состояние производится методами классов через косвенную адресацию.

## 5. Варианты контрольной работы

**Вариант 1**

1. Принципы объектно-ориентированного программирования.

2. Строки. Обработка строк. Стандартные функции работы со строками.

3. В справочной автовокзала хранится расписание движения автобусов. Для каждого рейса указаны:

- номер автобуса;
- тип автобуса;
- пункт назначения;
- дата отправления (дд/мм/гггг);
- время отправления;
- дата прибытия (дд/мм/гггг);
- время прибытия.

Выведите информацию о рейсах, которыми можно воспользоваться для прибытия в пункт назначения (вводится с клавиатуры) раньше заданного времени (вводится с клавиатуры).

4. Создайте в производном классе метод, определяющий:

$$t = \frac{2\cos\left(x - \frac{\pi}{6}\right)}{0{,}5 + \sin^2 y} \cdot \left(1 + \frac{z^2}{3 - \frac{z^2}{5}}\right).$$

**Вариант 2**

1. Синтаксические операторы: if, if/else, switch, for, while, do/while, break, continue.

2. Наследование: базовые и производные классы, защищенные члены класса, прямые и косвенные базовые классы.

3. В кассе аэровокзала имеется список авиапассажиров, в котором записаны:

- фамилия и инициалы пассажира;
- количество вещей багажа;
- общий вес багажа;
- номер рейса;
- дата вылета (дд/мм/гггг);
- время вылета;
- количество часов в полете;
- пункт назначения.

Определите фамилии пассажиров, летящих указанным рейсом (вводится с клавиатуры), суммарный вес багажа этих пассажиров, дату и время их прибытия в пункт назначения.

4. Создайте в производном классе метод, определяющий:

$$u = \frac{\sqrt[3]{8 + |x-y|^2 + 1}}{x^2 + y^2 + 2} - e^{|x-y|}(tg^2 z + 1)^x.$$

**Вариант 3**

1. Типы данных. Автоматическое и явное преобразование типов.

2. Классы. Область действия класс и доступ к членам класса.

3. Имеется список сотрудников фирмы, в котором записаны:
- фамилия;
- имя;
- отчество;
- дата рождения (дд/мм/гггг);
- место рождения.

Определите места рождения самого молодого и самого пожилого сотрудника. Определите, в какую пору года родилось больше всего сотрудников, и выведите их количество и фамилии.

4. Создайте в производном классе метод, определяющий:

$$v = \frac{1 + \sin^2(x+y)}{\left| x - \dfrac{2y}{1 + x^2 y^2} \right|} x^{|y|} + \cos^2\left( arctg \frac{1}{z} \right).$$

**Вариант 4**

1. Массивы: физическое и логическое определения, описание, доступ к элементам.

2. Наследование. Работа с экземплярами классов через ука-

затели, переопределение членов базового класса в производном классе.

3. Имеется список автолюбителей, в котором указаны следующие данные:

- марка автомобиля;
- год выпуска;
- номер двигателя;
- номер кузова;
- цвет автомобиля;
- номерной знак;
- Ф.И.О. владельца.

Определите фамилии владельцев, которые имеют автомобили указанной марки (вводится с клавиатуры) одинакового цвета. Вычислите количество автомобилей и выведите фамилии их владельцев, с момента выпуска которых прошло более, чем средний возраст всех автомобилей в списке.

4. Создайте в производном классе метод, определяющий:

$$w = \left|\cos x - \cos y\right|^{\left(1+2\sin^2 y\right)} \left(1 + z + \frac{z^2}{2} + \frac{z^3}{3} + \frac{z^4}{4}\right).$$

**Вариант 5**

1. Структуры. Описание. Создание структурных переменных. Доступ к элементам структуры.

2. Дружественные классы. Дружественные функции.

3. Имеется список автомашин, в котором записаны:

- марка автомашины;
- фамилия владельца;
- год выпуска;
- регистрационный номер;
- дата прохождения техосмотра (дд/мм/гггг).

Определите автомашины и их владельцев, не прошедших техосмотр (учесть, что автомашины старше 10 лет должны проходить техосмотр каждый год, остальные – 1 раз в 2 года). Подсчитайте количество автомашин, прошедших техосмотр в текущем году в указанном месяце (вводится с клавиатуры) и определите фамилии их владельцев.

4. Создайте в производном классе метод, определяющий:

$$\alpha = \ln\left(y^{-\sqrt{|x|}}\right)\left(x - \frac{y}{2}\right) + \sin^2 arctg\left(z\right).$$

**Вариант 6**

1. Указатели. Операции над указателями. Арифметика указателей.

2. Классы. Конструкторы и деструкторы.

3. Имеется список сотрудников фирмы, в котором записаны:

- фамилия;
- имя;
- отчество;
- дата рождения (дд/мм/гггг);
- телефон;
- адрес:
  - улица;
  - номер дома;
  - номер квартиры.

Определите фамилии и телефоны сотрудников, которые проживают на указанной улице (вводится с клавиатуры) в домах с четными номерами. Подсчитайте их средний возраст.

4. Создайте в производном классе метод, определяющий:

$$\beta = \sqrt{10\left(\sqrt[3]{x} + x^{y+2}\right)}\left(\arcsin^2 z - |x - y|\right).$$

**Вариант 7**

1. Ввод-вывод в C++. Библиотека iostream, манипуляторы, разделители.

2. Множественное наследование: синтаксис, порядок вызова конструкторов и деструкторов.

3. Имеется список сотрудников фирмы, в котором записаны:

- фамилия;
- имя;
- отчество;
- дата рождения (дд/мм/гггг);
- место рождения.

Определите Ф.И.О. сотрудников, которые родились в год быка (1901 – год быка). Определите, кто из сотрудников родился в указанную пору года (вводится с клавиатуры).

4. Создайте в производном классе метод, определяющий:

$$\gamma = 5arctg\left(x\right) - \frac{1}{4}\arccos(x)\frac{x + 3|x - y| + x^2}{|x - y|z + x^2}.$$

**Вариант 8**

1. Перегрузка имен функций. Встраиваемые функции.

2. Наследование. Использование конструкторов и деструкторов в производных классах.

3. Имеется список студентов, который содержит следующую информацию:

- Ф.И.О.;
- номер группы;
- оценка по физике;
- оценка по математике;
- оценка по информатике;
- дата сдачи последнего экзамена (дд/мм/гггг).

Определите Ф.И.О. студентов по группам в порядке убывания среднего балла, которые сдали последний экзамен до даты окончания сессии (вводится с клавиатуры).

4. Создайте в производном классе метод, определяющий:

$$\varphi = \frac{e^{|x-y|}\left|x-y\right|^{x+y}}{arctg\left(x\right)+arctg\left(z\right)} + \sqrt[3]{x^6+\ln^2 y}.$$

**Вариант 9**

1. Файлы последовательного доступа.

2. Перегрузка операторов.

3. В библиотеке имеется список книг. Каждая запись этого списка содержит:

- номер читательского билета;
- фамилии авторов;
- название книги;
- издательство;
- год издания;
- дата выдачи книги на руки (дд/мм/гггг);
- срок возврата.

Выведите информацию о книгах, срок возврата которых на указанную дату (вводится с клавиатуры) просрочен более чем на *N* дней (вводится с клавиатуры).

4. Создайте в производном классе метод, определяющий:

$$\psi = \left| x^{\frac{y}{x}} - \sqrt[3]{\frac{y}{x}} \right| + \left(y-x\right)\frac{\cos y - \dfrac{z}{(y-x)}}{1+\left(y-x\right)^2}.$$

**Вариант 10**

1. Файлы произвольного доступа.

2. Объекты. Создание объектов. Массивы объектов.

3. В магазине имеется список товаров, поступивших в продажу. Он содержит следующую информацию:

- группа товара (мясные изделия, молочные изделия и т. п.);
- наименование товара;
- изготовитель;
- дата изготовления (дд/мм/гггг);
- срок годности (количество дней).

Выведите информацию о товарах по группам товаров, срок годности которых на указанную дату (вводится с клавиатуры) просрочен.

4. Создайте в производном классе метод, определяющий:

$$b = y^{\sqrt[3]{|x|}} + \cos^3(y)\frac{|x-y|\left(1+\dfrac{\sin^2 z}{\sqrt{x+y}}\right)}{e^{|x-y|}+\dfrac{x}{2}}.$$

**Вариант 11**

1. Шаблоны функций. Встраиваемые функции.

2. Классы. Использование указателя this.

3. Для участия в конкурсе исполнителей необходимо заполнить следующую анкету:

- Ф.И.О.;
- дата рождения (дд/мм/гггг);
- название страны;
- класс музыкального инструмента (гитара, фортепиано, скрипка, виолончель).

Определите фамилии лауреатов конкурса по странам, чей возраст меньше, чем средний возраст всех лауреатов.

4. Создайте в производном классе метод, определяющий:

$$f = \frac{\sqrt[4]{y+\sqrt[3]{x-1}}}{|x-y|(\sin^2 z + tgz)}.$$

**Вариант 12**

1. Функции. Определения функций. Прототипы функций.

2. Наследование и иерархия классов.

3. Для участия в конкурсе на замещение вакантной должности сотрудника предприятия желающие подают следующую информацию:

- Ф.И.О.;
- год рождения;
- образование (среднее, среднее специальное, высшее);
- знание иностранных языков (английский, немецкий, французский; владею свободно, читаю и перевожу со словарем);
- владение компьютером;
- стаж работы;
- наличие рекомендаций.

Выведите список претендентов в соответствии с требованиями руководства фирмы (вводятся с клавиатуры).

4. Создайте в производном классе метод, определяющий:

$$c = 2^{\left(y^{x}\right)} + (3^{x})^{y} - \frac{y\left(arctg\left(z\right) - \frac{\pi}{6}\right)}{|x| + \frac{1}{y^{2}+1}}.$$

**Вариант 13**

1. Операции присваивания, инкремента и декремента, побитовые операции, логические операции, арифметические операции, операция следования (запятая).

2. Дружественные функции и классы.

3. Информация о сотрудниках предприятия содержит:

- Ф.И.О.;
- название отдела;
- должность;
- дату начала работы (дд/мм/гггг).

Определите фамилии сотрудников по отделам в порядке убывания общего трудового стажа, чей стаж работы превышает средний стаж в целом по предприятию.

4. Создайте в производном классе метод, определяющий:

$$b = y^{\sqrt[3]{|x|}} + \cos^{3}\left(y\right)\frac{|x-y|\left(1+\frac{\sin^{2} z}{\sqrt{x+y}}\right)}{e^{|x-y|} + \frac{x}{2}}.$$

**Вариант 14**

1. Многомерные массивы. Массивы указателей. Взаимосвязи между указателями и массивами.

2. Классы. Классы как элементы других классов.

3. В кассе аэровокзала имеется список авиапассажиров, в котором записаны:

- фамилия и инициалы пассажира;
- количество вещей багажа;
- общий вес багажа;
- номер рейса;
- дата вылета (дд/мм/гггг);
- время вылета;
- количество часов в полете;
- пункт назначения.

Определите, кто из пассажиров имеет багаж весом более *N* кг (вводится с клавиатуры), какими рейсами они летят, дату и время их прибытия в пункты назначения.

4. Создайте в производном классе метод, определяющий:

$$g = \frac{y^{x+1}}{\sqrt[3]{|y-2|}+3} + \frac{x+\dfrac{y}{2}}{2|x+y|}(x+1)^{-1/\sin z}.$$

**Вариант 15**

1. Структуры. Определения структур. Доступ к элементам структуры.

2. Абстрактные базовые классы и конкретные классы.

3. Имеется список сотрудников фирмы, в котором записаны:

- фамилия;
- имя;
- отчество;
- дата рождения (дд/мм/гггг);
- место рождения.

Определите фамилии, возраст, а также количество сотрудников, которые родились в определенном городе (вводится с клавиатуры).

4. Создайте в производном классе метод, определяющий:

$$h = \frac{x^{y+1}+e^{y-1}}{1+x|y-tgz|}(1+|y-x|) + \frac{|y-x|^2}{2} - \frac{|y-x|^3}{3}.$$

**Вариант 16**

1. Массивы. Объявление массивов. Передача массивов в функции.

2. Обработка исключительных ситуаций.

3. В кассе аэровокзала имеется список авиапассажиров, в котором записаны:

- фамилия и инициалы пассажира;
- количество вещей багажа;
- общий вес багажа;
- номер рейса;
- дата вылета (дд/мм/гггг);
- время вылета;
- количество часов в полете;
- пункт назначения.

Определите средний вес одной вещи в общем багаже и фамилии пассажиров, у которых средний вес одной вещи в их багаже отличается от среднего веса одной вещи в общем багаже не более чем на *N* кг (вводится с клавиатуры).

4. Создайте в производном классе метод, определяющий:

$$t = \frac{2\cos\left(x - \frac{\pi}{6}\right)}{0.5 + \sin^2 y}\left(1 + \frac{z^2}{3 - z^2/5}\right).$$

**Вариант 17**

1. Функции. Видимость переменных. Способы передачи аргументов. Аргументы по умолчанию.

2. Классы. Атрибуты доступа. Методы доступа к членам класса.

3. Информация об участниках спортивных соревнований содержит:

- наименование страны;
- название команды;
- Ф.И.О. игрока;
- игровой номер;
- дату рождения (дд/мм/гггг);
- рост;
- вес.

Определите для каждой команды количество, фамилии и

возраст участников, родившихся в указанную пору года (вводится с клавиатуры).

4. Создайте в производном классе метод, определяющий:

$$u = \frac{\sqrt[3]{8 + |x-y|^2 + 1}}{x^2 + y^2 + 2} - e^{|x-y|}(tg^2 z + 1)^x.$$

**Вариант 18**

1. Динамические структуры данных. Функции работы с динамической областью памяти.

2. Виртуальные функции. Виртуальные базовые классы.

3. Имеется список студентов, который содержит следующую информацию:

- Ф.И.О.;
- номер группы;
- оценка по физике;
- оценка по математике;
- оценка по информатике;
- дата сдачи последнего экзамена (дд/мм/гггг).

Определите Ф.И.О. студентов по группам, которые сдавали экзамены после даты окончания сессии (вводится с клавиатуры).

4. Создайте в производном классе метод, определяющий:

$$v = \frac{1 + \sin^2(x+y)}{\left| x - \frac{2y}{1 + x^2 y^2} \right|} x^{|y|} + \cos^2\left( arctg \frac{1}{z} \right).$$

**Вариант 19**

1. Строки. Обработка строк.

2. Применение полиморфизма. Раннее и позднее связывание.

3. В радиоателье хранятся квитанции о сданной в ремонт радиоаппаратуре. Каждая квитанция содержит следующую информацию:

- наименование группы изделий (телевизор, радиоприемник и т. п.);
- марку изделия;
- дату приемки в ремонт (дд/мм/гггг);
- состояние готовности заказа (выполнен, не выполнен).

Выведите информацию о состоянии заказов на указанную

дату (вводится с клавиатуры) по группам изделий.

4. Создайте в производном классе метод, определяющий:

$$\beta = \sqrt{10\left(\sqrt[3]{x} + x^{y+2}\right)}\left(\arcsin^2 z - |x - y|\right).$$

**Вариант 20**

1. Основные операции работы с выражениями и их приоритет. Условная операция. Операция sizeof.

2. Перегрузка операторов.

3. В библиотеке имеется список книг. Каждая запись этого списка содержит:

- фамилии авторов;
- название книги;
- издательство;
- год издания;
- дата выдачи книги на руки (дд/мм/гггг);
- срок возврата.

Выведите информацию о книгах, в названии которых встречается некоторое ключевое слово (вводится с клавиатуры). Определите на текущую дату (вводится с клавиатуры) названия книг, срок возврата которых просрочен.

4. Создайте в производном классе метод, определяющий:

$$w = |\cos x - \cos y|^{\left(1+2\sin^2 y\right)}\left(1 + z + \frac{z^2}{2} + \frac{z^3}{3} + \frac{z^4}{4}\right).$$

**Вариант 21**

1. Файловый ввод-вывод.

2. Объекты. Присваивание объектов. Передача объектов функциям. Объекты в качестве возвращаемого значения функций.

3. В библиотеке имеется список книг. Каждая запись этого списка содержит:

- номер книги;
- фамилии авторов;
- название книги;
- издательство;
- год издания;
- дата выдачи книги на руки (дд/мм/гггг);
- срок возврата.

Выведите информацию о книгах, изданных в одном и том же издательстве (вводится с клавиатуры) и вычислите их количество. Определите на текущую дату (вводится с клавиатуры) названия книг, до окончания срока возврата которых осталось не более *N* дней (вводится с клавиатуры).

4. Создайте в производном классе метод, определяющий:

$$\alpha = \ln\left(y^{-\sqrt{|x|}}\right)\left(x - \frac{y}{2}\right) + \sin^2 arctg\,(z).$$

**Вариант 22**

1. Типы данных. Преобразование типов. Объявление переменных. Описание констант.

2. Наследование. Управление доступом к базовому классу.

3. В магазине имеется список поступивших в продажу автомобилей. Каждая запись этого списка содержит:

- марку автомобиля;
- стоимость;
- расход топлива на 100 км;
- надежность (число лет безотказной работы);
- комфортность (отличная, хорошая, удовлетворительная).

Выведите перечень автомобилей, удовлетворяющих требованиям покупателя (вводятся с клавиатуры в виде некоторого интервала допустимых значений).

4. Создайте в производном классе метод, определяющий:

$$\gamma = 5arctg\,(x) - \frac{1}{4}\arccos(x)\frac{x + 3|x - y| + x^2}{|x - y|\,z + x^2}.$$

**Вариант 23**

1. Структуры. Описание и работа с ними. Создание структурных переменных.

2. Множественное наследование. Виртуальные базовые классы.

3. В предвыборной кампании проводится регистрация кандидатов в депутаты. Каждый кандидат, подавая заявление на регистрацию, указывает:

- номер округа, в котором он собирается баллотироваться;
- наименование партии, которую он представляет;
- дату рождения (дд/мм/гггг);

- профессию.

Определите: число поданных заявлений на регистрацию кандидатов каждой политической партии, средний возраст кандидатов от каждой политической партии, наиболее часто встречающуюся профессию кандидатов по каждой партии.

4. Создайте в производном классе метод, определяющий:

$$\varphi = \frac{e^{|x-y|}\,|x-y|^{x+y}}{arctg(x)+arctg(z)} + \sqrt[3]{x^6+\ln^2 y}.$$

**Вариант 24**

1. Функции. Способы передачи аргументов. Возвращаемые значения.

2. Перегрузка конструкторов. Создание и использование конструкторов копий.

3. Дана ведомость абитуриентов, сдавших вступительные экзамены в институт. В каждой строке данной ведомости записаны:

- Ф.И.О. абитуриента;
- постоянное место жительства;
- полученные им оценки по отдельным дисциплинам (например: физике, математике, литературе).

Определите количество абитуриентов, проживающих в заданном городе (вводится с клавиатуры) и сдавших экзамены со средним баллом не ниже *N* (вводится с клавиатуры). Выведите их фамилии в алфавитном порядке.

4. Создайте в производном классе метод, определяющий:

$$\psi = \left| x^{\frac{y}{x}} - \sqrt[3]{\frac{y}{x}} \right| + (y-x)\frac{\cos y - \dfrac{z}{(y-x)}}{1+(y-x)^2}.$$

**Вариант 25**

1. Перегрузка и шаблоны функций.

2. Инициализация объекта. Указатели, ссылки и массивы объектов.

3. У администратора железнодорожных касс хранится информация о свободных местах в поездах по всем направлениям на ближайшую неделю. Данная информация представлена в следующем виде:

- дата выезда (дд/мм/гггг);
- конечный пункт назначения;
- время отправления;
- число свободных купейных мест;
- число свободных плацкартных мест.

Оргкомитет международной конференции обращается к администратору с просьбой зарезервировать *m* (вводится с клавиатуры) мест до города *N* (вводится с клавиатуры) на указанную дату (вводится с клавиатуры). Выведите время отправления или сообщение о невозможности выполнить заказ в полном объеме.

4. Создайте в производном классе метод, определяющий:

$$b = y^{\sqrt[3]{|x|}} + \cos^3(y)\frac{|x-y|\left(1+\frac{\sin^2 z}{\sqrt{x+y}}\right)}{e^{|x-y|}+\frac{x}{2}}.$$

**Вариант 26**

1. Структурное программирование.
2. Конструкторы и деструкторы.
3. Информация об участниках спортивных соревнований содержит:

- наименование страны;
- название команды;
- Ф.И.О. игрока;
- игровой номер;
- дату рождения (дд/мм/гггг);
- рост;
- вес.

Выведите информацию о самой молодой, рослой и легкой командах.

4. Создайте в производном классе метод, определяющий:

$$f = \frac{\sqrt[4]{y+\sqrt[3]{x-1}}}{|x-y|(\sin^2 z + tgz)}.$$

**Вариант 27**

1. Рекурсия и итерация. Функции с пустыми списками параметров.

2. Одиночное наследование.

3. В кассе аэровокзала имеется список авиапассажиров, в котором записаны:

- фамилия и инициалы пассажира;
- количество вещей багажа;
- общий вес багажа;
- номер рейса;
- дата вылета (дд/мм/гггг);
- время вылета;
- количество часов в полете;
- пункт назначения.

Определите, имеются ли пассажиры, багаж которых состоит из одной вещи весом более *N* кг (вводится с клавиатуры). Определите их фамилии, номера рейсов, дату и время прилета в пункты назначения.

4. Создайте в производном классе метод, определяющий:

$$c = 2^{\left(y^x\right)} + (3^x)^y - \frac{y\left(arctg\left(z\right) - \frac{\pi}{6}\right)}{|x| + \frac{1}{y^2+1}}.$$

**Вариант 28**

1. Указатели и строки.
2. Система ввода-вывода C++.
3. Список студентов содержит следующую информацию:

- номер группы;
- Ф.И.О.,
- дата рождения (дд/мм/гггг);
- рост;
- вес.

Определите Ф.И.О. и количество студентов по группам, рост, вес и возраст которых одинаковые; рост и вес которых в списке являются уникальными.

4. Создайте в производном классе метод, определяющий:

$$g = \frac{y^{x+1}}{\sqrt[3]{|y-2|}+3} + \frac{x+\frac{y}{2}}{2|x+y|}(x+1)^{-1/\sin z}.$$

**Вариант 29**

1. Структуры и объединения.
2. Шаблоны классов.
3. На международной АТС информация о разговорах содержит:
- дату разговора (дд/мм/гггг);
- код и название города;
- время разговора;
- тариф;
- номер телефона в этом городе;
- номер телефона абонента.

Определите для каждого города общее время разговоров с ним и сумму на указанную дату (вводится с клавиатуры). Определите город, с абонентами которого наибольшее общее время разговоров.

4. Создайте в производном классе метод, определяющий:

$$h = \frac{x^{y+1} + e^{y-1}}{1 + x\left|y - tgz\right|}\left(1 + \left|y - x\right|\right) + \frac{\left|y - x\right|^2}{2} - \frac{\left|y - x\right|^3}{3}.$$

**Вариант 30**

1. Массивы. Описание, инициализация и обработка массивов.
2. Указатель this. Указатели на базовые и производные классы.
3. Каждая запись списка вакантных рабочих мест содержит:
- наименование организации;
- должность;
- квалификацию (разряд или образование);
- стаж работы по специальности;
- заработную плату;
- наличие социального страхования (да/нет);
- продолжительность ежегодного оплачиваемого отпуска.

Выведите список рабочих мест, удовлетворяющих требованиям клиента (вводятся с клавиатуры).

4. Создайте в производном классе метод, определяющий:

$$b = y^{\sqrt[3]{|x|}} + \cos^3(y)\frac{\left|x - y\right|\left(1 + \dfrac{\sin^2 z}{\sqrt{x + y}}\right)}{e^{|x-y|} + \dfrac{x}{2}}.$$

# 6. Основные теоретические сведения для выполнения практических заданий

**Ввод-вывод файлов.** C++ не предписывает никакой структуры файлу. Понятия, вроде «запись», не существуют в файлах языка C++. Следовательно, программист должен задавать структуру файлов в соответствии с требованиями прикладных программ.

Приложение, в котором предполагается использовать файлы произвольного доступа, должно буквально создать их. Для создания файлов произвольного доступа может быть использовано множество методов. Наиболее простым из них может являться требование, чтобы все записи в файле были одинаковой фиксированной длины.

Использование записей с фиксированной длиной дает возможность программе легко определять точное местоположение любой записи относительно начала файла как функцию от размера записи и ключа записи.

Пакет классов C++ работает с файловым вводом-выводом во многом так же, как он делает это со стандартным вводом-выводом.

Для реализации файлового ввода-вывода необходимо включить в программу заголовочный файл *fstream.* В нем определено несколько классов, включая классы *ifstream, ofstream* и *fstream*. Эти классы являются производными от классов *istream* и *ostream,* которые, в свою очередь, являются производными от класса *ios.*

В C++ файл открывается посредством его связывания с потоком. Имеются три типа потоков: ввода, вывода и ввода-вывода. Перед тем как открыть файл, нужно создать поток, поставить этот поток в соответствие определенному файлу (один из способов – воспользоваться методом *open(),* который является членом каждого из трех потоковых классов) и использовать этот поток как стандартные потоки ввода-вывода.

Приведем прототипы функции *open()* для каждого класса:

```
void ifstream::open(const char *имя_файла,
                    openmode режим=ios::in);
```

*void ofstream::open(const char *имя_файла,*
*openmode режим=ios::out|ios::trunc);*

*void fstream::open(const char *имя_файла,*
*openmode режим=ios::in|ios::out);*

Здесь *имя_файла* – имя файла, в которое может входить и спецификатор пути. Значение *режим* задает режим открытия файла. Оно должно быть значением типа *openmode*, которое является перечислением, определенным в классе *ios*. Значения возможных режимов открытия файла приведены в табл. 3.

Таблица 3

| Режим | Описание |
|---|---|
| ios::app | Записать все данные в конец файла |
| ios::ate | Переместиться в конец исходного открытого файла. Данные могут быть записаны в любое место файла |
| ios::in | Открыть файл для ввода |
| ios::out | Открыть файл для вывода (при открытии существующего файла его содержимое отбрасывается) |
| ios::trunc | Отбрасывать содержимое файла, если он существует (это также по умолчанию делается для ios::out) |

По умолчанию параметр режим функции *open()* устанавливается в значение, соответствующее типу открываемого потока, поэтому нет необходимости указывать его явно.

У классов *ifstream*, *ofstream* и *fstream* есть также конструкторы, которые открывают файл автоматически. Конструкторы имеют те же параметры, в том числе и задаваемые по умолчанию, что и функция *open()*. Поэтому возможна следующая запись:

*ifstream inFile (“test”);* //открытие файла для ввода

Для чтения данных из файла и записи в файл используются функции *read()* и *write()*, которые также являются членами потоковых классов соответственно для ввода и вывода.

Функция-элемент *write* класса *ostream* выводит фиксированное число байтов, начиная от заданного места в памяти, в заданный поток. Когда поток связан с файлом, данные пишутся, начиная с позиции в файле, определяемой при помощи указателя

позиции файла «put». Функция-элемент *read* класса *istream* вводит фиксированное число байтов из заданного потока в область памяти, начиная с указанного адреса. Если поток связан с файлом, байты вводятся начиная с позиции в файле, определенной при помощи указателя позиции файла «get».

Функция *write* ожидает первый аргумент типа *char **. Второй аргумент функции *write* является целым типа *size_t* и определяет число байтов, которые должны быть записаны.

Функция *read* класса *istream* может быть использована для последующего чтения фиксированного числа байтов, начиная от заданного места в памяти.

Программа обработки файлов произвольного доступа в редких случаях записывает единственное поле в файл. Обычно программы записывают за раз по одному объекту типа *struct* или *class*.

Следующий фрагмент демонстрирует открытие файла произвольного доступа, описание формата записи с помощью *struct* и запись данных на диск. Здесь задаются начальные значения 100 записям файла *credit.dat* в виде незаполненных записей типа *struct*, используя функцию *write*. Каждая незаполненная запись типа *struct* содержит номер счета 0, нулевую строку (представляемую при помощи пустых кавычек) в качестве фамилии и имени и 0.0 в качестве суммы. В файл первоначально записано столько незаполненных записей, сколько счетов предполагается хранить, для того чтобы в дальнейшем программа могла определять, содержит ли любая запись данные, или она является незаполненной.

```
...
struct clientData {
    int acctNum;
    char lastName[15];
    char firstName [10];
    float balance;
};

int main ()
{
    ofstream outCredit ("credit.dat", ios::out);
    if (! outCredit) {
            cerr << "Файл не может быть открыт"<<endl;
```

```
            exit(1);  //прототип в stdlib.h
        }
        clientData blankClient = {0, "", "",0.0};

        for (int i=1; i<=100; i++)
        outCredit.write((char *) &blankClient, sizeof(blankClient));
        return 0;
    }
```

Оператор *outCredit.write ((char*) &blankClient, sizeof(blankClient));* вызывает структуру *blankClient* размером *sizeof(blankClient)*, чтобы осуществить запись в файл *credit.dat*, связанный с объектом *outCredit* класса *ofstream*. Оператор *sizeof* возвращает размер в байтах объекта, заключенного в скобки.

Как класс *istream*, так и класс *ostream* содержат функции-элементы для позиционирования *указателя позиции файла* (это порядковый номер следующего байта в файле, который должен быть считан или записан). Этими функциями-элементами являются *seekg* (позиционировать для извлечения из потока) для класса *istream* и *seekp* (позиционировать для помещения в поток) для класса *ostream*. Любой объект класса *istream* имеет так называемый указатель «get», который показывает номер в файле очередного вводимого байта; а любой объект класса *ostream* имеет указатель «set», который показывает номер в файле очередного выводимого байта.

Аргумент функции *seekg* обычно является целым типа *long*. Второй аргумент, который может быть задан, показывает так называемое *направление позиционирования.* Направление позиционирования может быть *ios::beg* (по умолчанию) для позиционирования относительно начала потока, *ios::cur* – для позиционирования относительно текущей позиции в потоке и *ios::end* – для позиционирования относительно конца потока. Указатель позиции файла является целым числом, которое устанавливает позицию в файле как число байтов от начальной позиции в файле (иногда это называют *смещением* от начала файла).

Приведем несколько примеров позиционирования указателя позиции в файле для извлечения из потока:

```
// Позиционирование    fileObject на n-й байт
// полагаем ios::beg
```

*fileObject.seekg(n);*

// Позиционирование fileObject на *n* байтов вперед

*fileObject.seekg(n, ios::cur);*

// Позиционирование fileObject на *n*-й байт от конца файла *fileObject.seekg(y, ios::end);*

// Позиционирование fileObject на конец файла *fileObject.seekg(0, ios::end);*

Те же самые операции могут быть выполнены с помощью функции-элемента *seekp* класса *ostream.*

Функции-элементы *tellg* и *tellp* возвращают текущие позиции соответственно указателя взять из потока «get» и указателя поместить в поток «set». Следующий оператор присваивает переменной *location* типа *long* значение указателя «get».

```
location = fileObject.tellg();
```

Приведем пример программы, которая осуществляет последовательное чтение файла произвольного доступа, обновление данных уже записанных в файл, создание новых данных, которые размещаются в файле, удаление уже имеющихся в файле старых данных:

```
#include <stdafx.h>
#include <iostream.h>
#include <fstream.h>
#include <iomanip.h>
#include <stdlib.h>

struct clientData {
    int acctNum;
    char lastName[15];
    char firstName[10];
    float balance;
};

int enterChoice (void);
void textFile (fstream&);
void updateRecord (fstream&);
void newRecord (fstream&);
void deleteRecord (fstream&);
void outputLine (ostream&, clientData);
```

```
int main()

{
    fstream inOutCredit ("credit.dat", ios::in|ios::out);

    if (! inOutCredit) {
            cerr << "Файл не может быть открыт"<<endl;
            exit(1); //прототип в stdlib.h
    }
    int choice;
    while ((choice = enterChoice ())!=5) {
            switch (choice) {
            case 1:
                    textFile(inOutCredit);
                    break;
            case 2:
                    updateRecord(inOutCredit);
                    break;
            case 3:
                    newRecord(inOutCredit);
                    break;
            case 4:
                    deleteRecord(inOutCredit);
                    break;
            default:
                    cerr << "Incorrect choice" <<endl;
                    break;
            }
            inOutCredit.clear();
    }
    return 0;
}
//Приглашение выбрать раздел меню
int enterChoice(void)

{
    cout << endl<< "Viberite:"<< endl
         << "1 – создание текстового форматированного файла
         счетов"
         << endl
         << "с именем \"print.txt\* для печати"<< endl
         << "2 – izmenenie scheta"<< endl
         << "3 – dobavlenie novogo scheta"<< endl
```

```
            << "4 – udalenie scheta"<< endl
            << "5 – konec "<< endl<< "?";
    int menuChoice;
    cin>> menuChoice;
    return menuChoice;
}

//Создание форматированного текстового файла для печати
void textFile(fstream &readFromFile)
{
    ofstream outPrintFile ("print.txt", ios::out);

    if (! outPrintFile) {
            cerr << "Файл не может быть открыт"<<endl;
            exit(1);  //прототип в stdlib.h
    }
    outPrintFile<<setiosflags(ios::left)<<setw(6)<< "Счет"
            <<setw(16) << "Фамилия"<< setw(11)
            << "Имя"<<setiosflags(ios::right)
            <<setw(10)<<"Баланс"<< endl;
    readFromFile.seekg(0);

    clientData client;
    readFromFile.read((char*)&client, sizeof(client));

    while (!readFromFile.eof()) {
            if (client.acctNum!=0)
                    outputLine (outPrintFile, client);

            readFromFile.read ((char*) &client, sizeof (client));
    }
}
//Изменение баланса счета

void updateRecord (fstream &updateFile)
{
    int account;

    do {
            cout<<"Vvedite schet (1-100):";
            cin >>account;
            clientData client;
    } while (account <1|| account>100);
```

```
    updateFile.seekg((account-1)* sizeof(clientData));

    clientData client;
    updateFile.read ((char*) &client, sizeof(clientData));

    if (client.acctNum !=0) {
            outputLine (cout, client);
            cout<<endl<<"Введите расход (+) или доплату (-):";

            float transaction;
            cin>> transaction;
            client.balance += transaction;
            outputLine (cout, client);
            updateFile.seekp((account-1)* sizeof(client));
            updateFile.write((char*)&client, sizeof(client));
    }
    else
            cerr << "Счет №"<< account
            << "не заполнен."<<endl;
}
//Создание и вставка новой записи

void newRecord (fstream &insertInFile)
{

    cout<<"Введите новый номер счета (1-100):";
    int account;
    cin >>account;
    insertInFile.seekg((account-1)+sizeof(clientData));

    clientData client;
    insertInFile.read ((char*)&client, sizeof(client));

    if (client.acctNum == 0) {
            cout<< "Введите фамилию, имя, баланс" <<endl <<
            "?";
            cin>> client.lastName>>client.firstName>> cli-
            ent.balance;
            client.acctNum = account;
            insertInFile.seekp((account-1)* sizeof(clientData));
            insertInFile.write((char*)&client, sizeof(clientData));
    }
    else
```

```
            cerr << "Счет №"<< account
            << "уже содержит информацию."<<endl;
}

//Удаление существующей записи

void deleteRecord (fstream &deleteFromFile)
{

    cout<<"Введите  номер счета для удаления (1–100):";

    int account;
    cin >>account;
    deleteFromFile.seekg((account-1)+sizeof(clientData));

    clientData client;
    deleteFromFile.read ((char*)&client, sizeof(client));

    if (client.acctNum != 0) {
            clientData blankClient = {0, "","",0.0};

            deleteFromFile.seekp((account-1)* sizeof(client));
            deleteFromFile.write((char*)&client, sizeof(client));
            cout<<"Счет №"<<account<<"удален."<<endl;
    }
    else
            cout<<"Счет №"<<account<<"пустой."<<endl;
}
//Вывод строки с информацией о клиенте

void outputLine(ostream &output, clientData c)
{
    cout << setiosflags (ios::left)<< setw(6) <<c.acctNum
            << setw(16) << c.lastName << setw(11) << c.firstName
            << setw(10)<<setprecision(2)
            << setiosflags (ios::showpoint| ios::right)
            << c.balance<< endl;
}
```

**Наследование и механизм виртуальных функций.** *Наследование (inheritance)* является одной из базовых концепций объектно-ориентированного программирования и подразумевает использование принципа иерархии и абстрагирования объектов.

Наследование – это процесс, посредством которого один объект может наследовать основные свойства другого объекта и добавлять к ним черты, характерные только для него.

Часто объекты одного класса на самом деле являются также и объектами другого класса. Прямоугольник является также и четырехугольником (как и квадрат, и параллелограмм, и трапеция). Таким образом, можно сказать, что класс Rectangle (прямоугольник) является *наследником* класса Quadrilaterial (четырехугольник). В этом контексте класс Quadrilaterial является *базовым классом,* а класс Rectangle – *производным классом.* Прямоугольник является специальным типом четырехугольника, но неверно утверждать, что четырехугольник является прямоугольником.

Наследование называется простым (одиночным), когда каждый класс порождается только от одного базового класса. В C++ наследование может быть множественным. Множественное наследование означает, что производный класс наследует элементы нескольких базовых классов.

При порождении класса из базового класса этот базовый класс может наследоваться как *public (открытый), protected (защищенный)* или *private (закрытый).*

При порождении класса как *public* открытые элементы базового класса становятся открытыми элементами производного класса, а защищенные элементы базового класса – защищенными элементами производного класса. Закрытые элементы базового класса никогда не бывают доступны для производного класса.

При защищенном наследовании открытые и защищенные элементы базового класса становятся защищенными элементами производного класса. При закрытом наследовании открытые и защищенные элементы базового класса становятся закрытыми элементами производного класса.

Когда один класс наследуется другим, используется следующая основная форма записи:

*class имя_произв_класса : сп_доступа имя_баз_класса {...};*

*сп_доступа* – это одно из ключевых слов *public, protected* или *private.* Технически спецификатор доступа не обязателен. Если спецификатор доступа не указан и производный класс определен с ключевым словом *class*, то базовый класс по умолчанию наследуется как закрытый.

Если у базового и производного классов имеются конструкторы и деструкторы, то конструкторы выполняются в порядке наследования, а деструкторы – в обратном порядке.

Поскольку производный класс наследует элементы базового класса, то при создании объекта производного класса должен быть вызван конструктор базового класса для задания начальных значений элементам базового класса, содержащимся в объекте производного класса. В конструкторе производного класса при явном вызове конструктора базового класса может быть предусмотрен список инициализаторов элементов, в противном случае конструктор производного класса будет неявно вызывать конструктор базового класса с умолчанием.

Для того чтобы осуществить передачу аргументов для конструктора базового класса, необходимо воспользоваться следующей записью:

*констр_произв_класса (аргументы) : констр_баз_класса(аргументы)*

Количество и тип аргументов для конструкторов производных и базового класса могут не совпадать.

Отметим еще одну особенность, связанную с использованием указателей на базовый и производный классы. Если в программе определен указатель на базовый класс, то его можно использовать для доступа к компонентам объектов базового класса, а также для доступа к компонентам базового класса, которые наследуются в производном классе. Если такой указатель указывает на экземпляр производного класса, то с помощью этого указателя нельзя получить доступ к тем компонентам, которые описаны только внутри производного класса. Если же описан указатель на производный класс, то при помощи такого указателя можно осуществлять доступ к открытым компонентам данного класса, в том числе и к компонентам базового класса, которые открыто наследуются в производном классе. Нельзя использовать указатель производного класса для доступа к компонентам объектов базового класса.

Виртуальная функция (*virtual function*) является членом класса и объявляется с ключевым словом *virtual* внутри базового класса и переопределяется в производном классе. По существу, виртуальная функция реализует идею «один интерфейс, множество методов», которая лежит в основе полиморфизма. При пе-

реопределении виртуальной функции в производном классе, ключевое слово *virtual* не требуется. Переопределяемая виртуальная функция должна иметь точно такой же тип параметров, то же их число и такой же тип возвращаемого значения, что и виртуальная функция базового класса (иначе она становится перегружаемой функцией и ее виртуальная природа теряется).

Виртуальные функции имеют иерархический порядок наследования. Если виртуальная функция не переопределяется в производном классе, то используется версия функции, определенная в базовом классе.

Виртуальная функция может вызываться так же, как и любая другая функция-член. Однако наиболее интересен вызов виртуальной функции через указатель, благодаря чему поддерживается динамический полиморфизм. Тип объекта производного класса, на который ссылается указатель базового класса, определяет ту версию виртуальной функции, которая будет выполняться. При этом определение конкретной версии виртуальной функции имеет место в процессе выполнения программы. Этот процесс является реализацией принципа динамического полиморфизма.

О классе, который содержит виртуальную функцию, говорят как о полиморфном классе (*polymorphic class*).

Как правило, на практике в базовом классе приводят только прототип виртуальной функции, которая приравнивается нулю. Это сообщает компилятору, что в базовом классе не существует тела функции. Такая функция называется чистой виртуальной функцией (*pure virtual function*), а класс, ее содержащий, – абстрактным. Объекты абстрактных классов создавать нельзя, однако можно определять указатели. Чистая виртуальная функция обязательно должна быть переопределена в производном классе.

Рассмотрим вышеперечисленные свойства на примере:

```
#include <iostream.h>
class A                                  //базовый класс
{
    int x;
    public:
    A(int n){x=n;}
    virtual void print(){cout<<"x="<<x<<endl;}
};
class B:public A                 //производный класс
```

```
{
    int y;
    public:
    B(int m, int n):A(n+3 ) //конструктор со списком инициализации
            {y=m;}
    void print(){cout<<"y="<<y<<endl;}
};

main()
{
    A obj1(1);
    B obj2(5,6);
    A *ptr;
    ptr=&obj1;
    ptr->print(); // x=1
    ptr=&obj2;
    ptr->print(); // y=5
    return 0;
}
```

## 7. Курсовой проект

Учебным планом специальности 1-08 01 01-07 «Профессиональное обучение. (Информатика)» для студентов безотрывной формы обучения предусмотрено выполнение курсового проекта по дисциплине «Конструирование программ и языки программирования».

Цель курсового проекта – проявление студентом в полной мере своих знаний, умений, навыков и творческих возможностей в области анализа, проектирования и программирования сложных задач и получение навыков оформления технической, графической, программной документации.

***Примерный перечень тем курсовых проектов:***

1. Автоматизация решения систем линейных уравнений.
2. Автоматизация построения графиков функций.
3. Автоматизация поиска оптимального пути в системе городов.
4. Автоматизация поиска пути в лабиринте.
5. Анализ быстродействия методов внутренней сортировки.
6. Архиватор текстов.
7. Текстовый редактор.

8. Графический редактор.
9. Автоматизация решения задач линейного программирования симплекс-методом.
10. Система психологического тестирования.
11. Обработка строковых данных и текстовых файлов.
12. Генератор кроссвордов.
13. Оптимизация работы параллельных процессоров.
14. Демонстрационная программа выполнения геометрических преобразований трехмерных объектов.
15. Расчет и психоанализ врожденных характеристик человека.
16. Информационно-справочная система музыкальных произведений.

***Состав курсового проекта:***

- текстовая документация (пояснительная записка);
- графическая документация (различные схемы, диаграммы);
- отлаженное, протестированное программное средство.

Объем пояснительной записки (ПЗ) должен быть 25–40 страниц печатного текста (без учета текста программы) и оформлять ее следует в соответствии с требованиями, изложенными в [12].

Графическая часть проекта должна быть представлена в виде диаграмм, разработанных на этапах объектно-ориентированного проектирования и программирования с помощью Microsoft Visio или CASE-средства Rational Rose на основе UML (универсальный язык моделирования). Графическая документация оформляется в виде приложения (приложений) на листах формата А4. Правила разработки диаграмм изложены в [4].

Разработку программного средства необходимо выполнять, используя концепции объектно-ориентированного проектирования и программирования, на языке программирования С++ под ОС Windows.

При объектно-ориентированном подходе к разработке программы пояснительная записка должна иметь следующий состав:

Введение.

1. Постановка задачи *(описание предметной области задачи, обзор существующих решений, требования к программе, функции программы, входные и выходные данные)*.

2. Вычислительная система *(требования к аппаратным и программным средствам, выбор и обоснование операционной системы, языка программирования и среды реализации)*.

3. Объектно-ориентированное проектирование *(выделение сущностей предметной области задачи, разработка диаграмм вариантов использования, диаграмм классов, диаграмм деятельности, диаграмм последовательности и др.)*.

4. Объектно-ориентированное программирование *(описание исходного кода программы: разработанных классов, методов, структуры программы)*.

5. Описание программного средства *(описание аппаратных и программных средств, необходимых для успешной работы программы, руководство пользователя)*.

Заключение.

Литература.

Приложения *(диаграммы, листинг программы, результаты работы программы, копии экранных форм)*.

Примечание. Каждый раздел ПЗ может быть представлен подразделами, количество и содержание которых определяется разработчиком курсового проекта.

Темы курсовых проектов предлагаются преподавателем индивидуально каждому студенту и утверждаются заведующим кафедрой информатики. Документально это оформляется в виде листа технического задания на курсовое проектирование установленного образца.

В ходе выполнения курсового проекта студент может получить консультацию у преподавателя-руководителя проекта.

К защите студент должен представить программное средство, пояснительную записку (на листах формата А4), графическую документацию, разработанные в соответствии с техническим заданием на курсовое проектирование и вышеприведенными требованиями.

**Рекомендуемая литература**


***Основная***

1. Буч, Г. Объектно-ориентированное проектирование с примерами приложений на C++ / Г. Буч. – М. : Бином, 1999.
2. Голубь, Н. Г. Искусство программирования на Ассемблере / Н. Г. Голубь. – М. ; СПб. ; Киев : Диасофт, 2002.
3. Дейтел, Х. Как программировать на C++ / Х. Дейтел, П. Дейтел ; пер. с англ. – М. : ЗАО «Издательство БИНОМ», 2000.
4. Леоненков, А. Самоучитель UML / А. Леоненков. – СПб. : BHV – Санкт-Петербург, 2004.
5. Прата, С. Язык программирования C++ : лекции и упражнения / С. Прата ; пер. с англ. – М. : Диасофт, 2005.
6. Шилдт, Г. Самоучитель C++ / Г. Шилдт ; пер. с англ. – 3-е изд. – СПб. : BHV – Санкт-Петербург, 2000.
7. Юров, В. И. ASSEMBLER / В. И. Юров. – СПб. : Питер, 2002.

***Дополнительная***

8. Бондарев, В. М. Основы программирования / В. М. Бондарев, В. И. Рублинецкий, Е. Г. Качко. – Харьков : Фолио ; Ростов н/Д : Феникс, 1997.
9. Вирт, Н. Алгоритмы и структуры данных / Н. Вирт. – СПб. : Невский Диалект, 2001.
10. Касаткин, А. И. От Turbo C к Borland C++ / А. И. Касаткин, А. Н. Вальвачев. – Мн. : Выш. шк., 1992.
11. Кнут, Д. Искусство программирования для ЭВМ : в 3 т. / Д. Кнут. – М. : Мир, 1976.
12. Оформление курсовых и дипломных проектов : метод. указания / сост. И. М. Снежкова. – Мн. : МГВРК, 2003.
13. Фаронов, В. В. Turbo Pascal 7.0 : практика программирования / В. В. Фаронов. – Нолидж, 1997.

## Содержание

Учебное издание

**КОНСТРУИРОВАНИЕ ПРОГРАММ
И ЯЗЫКИ ПРОГРАММИРОВАНИЯ**

Учебная программа, методические указания
и контрольные задания
для студентов безотрывной формы обучения
специальности 1-08 01 01-07
«Профессиональное обучение. (Информатика)»

Составители:
**Бельчик** Марина Анатольевна
**Заневская** Валентина Александровна

Зав. ред.-издат. отд. О. П. Козельская
Редактор Г. Л. Говор
Компьютерная верстка А. П. Пучек

План издания 2007 г. (поз. 2)

Изд. лиц. № 02330/0131735 от 17.02.2004.
Подписано в печать 19.04.2007. Формат 60×84 $^{1}/_{16}$.
Бумага писчая. Гарнитура Таймс. Печать ризографическая.
Усл. печ. л. 3,02. Уч.-изд. л. 1,64. Тираж 60 экз. Заказ 73.

Издатель и полиграфическое исполнение Учреждение образования
«Минский государственный высший радиотехнический колледж»
220005, г. Минск, пр-т Независимости, 62.

ISBN 978-985-6754-91-6