**ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ РЕСПУБЛИКИ КРЫМ**

**«КРЫМСКИЙ ИНЖЕНЕРНО-ПЕДАГОГИЧЕСКИЙ УНИВЕРСИТЕТ»**

**Кафедра прикладной информатики**

|  |  |
| --- | --- |
| «СОГЛАСОВАНО»  Руководитель ОПОП  \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Ильясова Ф.С.  «\_\_\_»\_\_\_\_\_\_\_\_2017 года | «УТВЕРЖДАЮ»  Заведующий кафедрой  \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Сейдаметова З.С.  «\_\_\_»\_\_\_\_\_\_\_\_2017 года |

**РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ**

**«Б1.Б.9 ИФОРМАТИКА И ПРОГРАММИРОВАНИЕ»**

Направление подготовки

**09.03.03 Прикладная информатика**

Профиль

**Прикладная информатика**

Факультет экономики, менеджмента и информационных технологий

Симферополь, 2017

Рабочая программа дисциплины «Информатика и программирование» для бакалавров направления подготовки 09.03.03 Прикладная информатика профиля «Прикладная информатика» составлена на основании ФГОС ВО, утвержденного приказом Министерства образования и науки РФ от 12.03.2015 г. № 207, и учебного плана по направлению подготовки 09.03.03 Прикладная информатика, утвержденного Ученым советом ГБОУВО РК КИПУ от 24.04.17г., протокол № 12.

Составитель рабочей программы к.пед.н.,ст. преп. Ильясова Ф.С.

ст.преп. Сейтвелиева С.Н.

Рабочая программа утверждена на кафедре прикладной информатики

Протокол № 1 от 30 августа 2017 г.

Заведующий кафедрой \_\_\_\_\_\_\_\_\_\_\_З.С. Сейдаметова

Рабочая программа одобрена и утверждена на заседании УМК факультета экономики, менеджмента и информационных технологий

Протокол №1 от 05 сентября 2017 г.

Председатель УМК \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ К.М. Османов

Рабочая программа переутверждена на заседании кафедры прикладной информатики

Протокол № \_\_\_\_\_\_\_ от \_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_\_\_г.

Заведующий кафедрой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Содержание**

[1. Перечень планируемых результатов обучения по дисциплине, соотнесенных с планируемыми результатами освоения образовательной программы 5](#_Toc468032255)

[3. Объем дисциплины в зачетных единицах с указанием количества академических часов, выделенных на контактную работу обучающихся с преподавателем (по видам учебных занятий) и на самостоятельную работу 6](#_Toc468032256)

[4. Содержание дисциплины (структурированное по темам с указанием отведенного на них количества академических или астрономических часов и видов учебных занятий) 7](#_Toc468032257)

[4.2. Тематический план лекций 10](#_Toc468032258)

[4.3. Темы лабораторных занятий 13](#_Toc468032259)

[5. Перечень учебно-методического обеспечения для самостоятельной работы обучающихся по дисциплине 14](#_Toc468032260)

[5.1. Содержание самостоятельной работы студентов по дисциплине 14](#_Toc468032261)

[6. Фонд оценочных средств для проведения промежуточной аттестации обучающихся по дисциплине 15](#_Toc468032262)

[6.1. Перечень компетенций с указанием этапов их формирования в процессе освоения образовательной программы 15](#_Toc468032263)

[6.2. Описание показателей и критериев оценивания компетенций на различных этапах их формирования, описание шкал оценивания 16](#_Toc468032264)

[6.3. Типовые контрольные задания или иные материалы, необходимые для оценки знаний, умений, навыков и (или) опыта деятельности, характеризующих этапы формирования компетенций в процессе освоения образовательной программы 17](#_Toc468032265)

[Оформление комплекта заданий для контрольной работы 19](#_Toc468032266)

[6.4. Методические материалы, определяющие процедуры оценивания знаний, умений, навыков и (или) опыта деятельности, характеризующих этапы формирования компетенций. 19](#_Toc468032267)

[6.5. Итоговая рейтинговая оценка текущей и промежуточной аттестации студента по дисциплине 19](#_Toc468032268)

[7. Перечень основной и дополнительной учебной литературы, необходимой для освоения дисциплины 22](#_Toc468032269)

[8. Перечень ресурсов информационно-телекоммуникационной сети Интернет, необходимых для освоения дисциплины 23](#_Toc468032270)

[9. Методические указания для обучающихся по освоению дисциплины 23](#_Toc468032271)

[9.1. Общие рекомендации по самостоятельной работе студентов 23](#_Toc468032272)

[9.2 Методические рекомендации по освоению лекционного материала 24](#_Toc468032273)

[9.3. Методические рекомендации по подготовке к лабораторным занятиям 25](#_Toc468032274)

[10. Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине (включая перечень программного обеспечения и информационных справочных систем (при необходимости)) 25](#_Toc468032275)

[11. Описание материально-технической базы, необходимой для осуществления образовательного процесса по дисциплине 25](#_Toc468032276)

[12. Методические материалы к РПД 25](#_Toc468032277)

[Методические рекомендации по подготовке к лабораторным занятиям 25](#_Toc468032278)

[Вопросы для подготовки к экзамену по дисциплине 48](#_Toc468032279)

[Экзаменационные билеты 50](#_Toc468032280)

[Оформление комплекта заданий для контрольной работы 52](#_Toc468032281)

[Опорный конспект лекций 59](#_Toc468032282)

**Рабочая программа дисциплины**

**УЧЕБНОЙ ДИСЦИПЛИНЫ**

**«**ИНФОРМАТИКА И ПРОГРАММИРОВАНИЕ**»**

**программы бакалавриата по направлению подготовки 09.03.03**

# 1. Перечень планируемых результатов обучения по дисциплине, соотнесенных с планируемыми результатами освоения образовательной программы

*Цель и задачи изучения дисциплины*

**Цель** изучения дисциплины «Информатика и программирование» студентов направления подготовки 09.03.03 «Прикладная информатика» состоит в подготовке базиса для овладения специальностью инженера-программиста и развитии умений поиска наиболее эффективного решения задачи с последующей программной реализацией этого решения.

**Задачи дисциплины** «Информатика и программирование» следующие:

* сформировать умение разрабатывать алгоритм и записывать его в разной форме, научить владеть терминологией;
* сформировать умение выделять и описывать объекты задачи и их взаимодействие;
* сформировать умение понимать семантику основных управляющих структур программного кода, сформировать навыки записывать программный код;
* сформировать умение эффективно использовать интегрированную программную среду в ходе решения задачи.

В результате изучения дисциплины (учебного курса) студент формирует и демонстрирует следующие компетенции:

* способность разрабатывать, внедрять и адаптировать прикладное программное обеспечение (ПК-2)
* способность программировать приложения и создавать программные прототипы решения прикладных задач (ПК-8)
* способность проводить тестирование компонентов программного обеспечения ИС (ПК-12).

Сформированность указанных компетенций определяется тем, что студент должен

**знать:**

* понятие алгоритма и свойств, различные записи формы алгоритма;
* понятие двоичной системы исчисления и алгоритм перевода числа из десятичной системы в двоичную и обратно;
* стандартные и пользовательские типы данных языка С++, основные понятия объектно-ориентированного программирования;
* синтаксис и семантику управляющих конструкций программы на языке С++, понятие интегрированной среды программирования и её основные компоненты.

**уметь**:

* выделять входные и выходные данные задачи, записывать алгоритмы в разной форме, записывать и компилировать исходный код на языке С++;
* использовать отладчик для поиска логических ошибок;
* использовать компоненты библиотеки STL, разделять исходный код на логически независимые составляющие;
* выделять классы, их свойства и методы, разрабатывать GUI.

**владеть:**

* навыками процедурного программирования, базовыми навыками объектно-ориентированного программирования;
* компиляцией многофайловых программ;
* схемами хранения данных;
* навыками обобщенного программирования (библиотекой STL), библиотекой Qt для разработки GUI приложений.

**2. Место дисциплины в структуре образовательной программы**

Данная дисциплина относится к обязательным дисциплинам. Дисциплины, учебные курсы, на освоении которых базируется данная дисциплина (учебный курс) – «Дискретная математика», «Программирование для начинающих», «Введение в специальность» и т.д.

Дисциплины, учебные курсы, для которых необходимы знания, умения, навыки, приобретаемые в результате изучения данной дисциплины (учебного курса) – «Объектно-ориентированное программирование», «Алгоритмы и структуры данных», «Системное программирование», «Язык программирование Java».

# 3. Объем дисциплины в зачетных единицах с указанием количества академических часов, выделенных на контактную работу обучающихся с преподавателем (по видам учебных занятий) и на самостоятельную работу

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Семестр | Общее количество часов | Количество зачетных единиц | Контактные часы | | | | | кср | с/р | Итоговый контроль (экзамен, зачет) |
| Всего | л | п | с | л/р |
| ДФО | | | | | | | | |  | |
| 1 | 144 | 4 | 68 | 30 |  |  | 34 | 4 | 49 | экзамен (27) |
| 2 | 144 | 4 | 68 | 30 |  |  | 34 | 4 | 49 | экзамен (27) |
| 3 | 144 | 4 | 68 | 30 |  |  | 34 | 4 | 49 | экзамен (27) |
| ЗФО | | | | | | | | |  | |
| 1 | 144 | 4 | 16 | 8 |  |  | 8 |  | 119 | экзамен (9) |
| 2 | 144 | 4 | 16 | 8 |  |  | 8 |  | 119 | экзамен (9) |
| 3 | 144 | 4 | 14 | 6 |  |  | 8 |  | 121 | экзамен (9) |

# 4. Содержание дисциплины (структурированное по темам с указанием отведенного на них количества академических или астрономических часов и видов учебных занятий)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Наименования тем | Количество часов | | | | | | | | | | | | | | Формы текущего контроля |
| дневная форма | | | | | | | заочная форма | | | | | | |
| всего | в том числе | | | | | | всего | в том числе | | | | | |
| л | п | с | лаб | К | СР | л | п | с | лаб | К | СР |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
| **1 семестр** | | | | | | | | | | | | | | | |
| **Раздел 1. Введение в императивное Информатика и программирование** | | | | | | | | | | | | | | | |
| Тема 1. Понятие программы, компилятора, исходного и исполняемого кода | 11 | 3 |  |  | 2 | 2 | 4 | 16 | 2 |  |  | 2 | 2 | 10 | контрольная работа/тестовый контроль |
| Тема 2. Понятие алгоритма. Запись алгоритма на естественном и формальном языках | 11 | 3 |  |  | 2 | 2 | 4 | 16 | 2 |  |  | 2 | 2 | 10 | контрольная работа/тестовый контроль |
| Тема 3. Императивная парадигма программирования. Структура программы на языке С++ | 11 | 3 |  |  | 2 | 2 | 4 | 16 | 2 |  |  | 2 | 2 | 10 | контрольная работа/тестовый контроль |
| Тема 4. Основы работы в среде программирования | 11 | 3 |  |  | 2 | 2 | 4 | 17 | 2 |  |  | 2 | 3 | 10 | контрольная работа/тестовый контроль |
| Тема 5. Типы данных в языке С++ | 11 | 3 |  |  | 2 | 2 | 4 | 10 |  |  |  |  |  | 10 | контрольная работа/тестовый контроль |
| Тема 6. Арифметические и логические выражения | 11 | 3 |  |  | 2 | 2 | 4 | 10 |  |  |  |  |  | 10 | контрольная работа/тестовый контроль |
| Тема 7. Управляющие структуры языка С++ | 11 | 3 |  |  | 2 | 2 | 4 | 10 |  |  |  |  |  | 10 | контрольная работа/тестовый контроль |
| Тема 8. Отладка и тестирование программ | 11 | 3 |  |  | 2 | 2 | 4 | 10 |  |  |  |  |  | 10 | контрольная работа/тестовый контроль |
| Тема 9. Побитовые операции | 10 | 2 |  |  | 2 | 2 | 4 | 10 |  |  |  |  |  | 10 | контрольная работа/тестовый контроль |
| Тема 10. Функции | 10 | 2 |  |  | 2 | 2 | 4 | 10 |  |  |  |  |  | 10 | контрольная работа/тестовый контроль |
| **2. Элементы объектно-ориентированного программирования** | | | | | | | | | | | | | | |  |
| Тема 1. Понятие класса и объекта, методы класса | 11 | 2 |  |  | 2 | 4 | 3 | 10 |  |  |  |  |  | 10 | контрольная работа/тестовый контроль |
| Тема 2. Конструктор и деструктор класса | 9 | 2 |  |  | 2 | 3 | 2 | 9 |  |  |  |  |  | 9 | контрольная работа/тестовый контроль |
| **Всего за семестр** | **140** | **32** |  |  | **36** | **27** | **45** | **144** | **8** |  |  | **8** | **9** | **119** |  |
| Форма контроля | экзамен  КСР – 4 часа | | | | | | | экзамен | | | | | | | экзамен |
| **2 семестр** | | | | | | | | | | | | | | |  |
| **Раздел 3. Пользовательские структуры данных** | | | | | | | | | | | | | | |  |
| Тема 1. Пользовательские типы | 14 | 3 |  |  | 3 | 3 | 5 | 18 | 2 |  |  | 2 | 2 | 12 | контрольная работа/тестовый контроль |
| Тема 2. Указатели | 14 | 3 |  |  | 3 | 3 | 5 | 18 | 2 |  |  | 2 | 2 | 12 | контрольная работа/тестовый контроль |
| Тема 3. Динамическое выделение памяти | 14 | 3 |  |  | 3 | 3 | 5 | 18 | 2 |  |  | 2 | 2 | 12 | контрольная работа/тестовый контроль |
| Тема 4. Списки | 14 | 3 |  |  | 3 | 3 | 5 | 19 | 2 |  |  | 2 | 3 | 12 | контрольная работа/тестовый контроль |
| Тема 5. Строки в С++ | 14 | 3 |  |  | 3 | 3 | 5 | 12 |  |  |  |  |  | 12 | контрольная работа/тестовый контроль |
| **Раздел 4. Императивное Информатика и программирование** | | | | | | | | | | | | | | |  |
| Тема 1. Область видимости и время жизни | 13 | 3 |  |  | 3 | 3 | 4 | 12 |  |  |  |  |  | 12 | контрольная работа/тестовый контроль |
| Тема 2. Функции: параметры-ссылки и параметры-указатели | 14 | 3 |  |  | 4 | 3 | 4 | 12 |  |  |  |  |  | 12 | контрольная работа/тестовый контроль |
| Тема 3. Раздельная компиляция. Заголовочные файлы | 13 | 3 |  |  | 4 | 2 | 4 | 12 |  |  |  |  |  | 12 | контрольная работа/тестовый контроль |
| Тема 4. Рекурсия и итерация | 14 | 4 |  |  | 4 | 2 | 4 | 2 |  |  |  |  |  | 12 | контрольная работа/тестовый контроль |
| Тема 5 Шаблоны функций | 14 | 4 |  |  | 4 | 2 | 4 | 11 |  |  |  |  |  | 11 | контрольная работа/тестовый контроль |
| **Всего за семестр** | 140 | 32 |  |  | 36 | 27 | 45 | 144 | 8 |  |  | 8 | 9 | 119 |  |
| Форма контроля | экзамен  КСР – 4 часа | | | | | | | экзамен | | | | | | | экзамен |
| **семестр 3** | | | | | | | | | | | | | | |  |
| **Раздел 5. Перегруженные функции** | | | | | | | | | | | | | | |  |
| Тема 1. Перегруженные функции. Объявление перегруженных функций | 13 | 3 |  |  | 3 | 3 | 4 | 18 | 2 |  |  | 2 | 2 | 12 | контрольная работа/тестовый контроль |
| Тема 2. Преобразование типов параметров перегруженных функций | 13 | 3 |  |  | 3 | 3 | 4 | 18 | 2 |  |  | 2 | 2 | 12 | контрольная работа/тестовый контроль |
| Тема 3. Обработка исключений | 13 | 3 |  |  | 3 | 3 | 4 | 18 | 2 |  |  | 2 | 2 | 12 | контрольная работа/тестовый контроль |
| Тема 4. Директивы препроцессора | 13 | 3 |  |  | 3 | 3 | 4 | 17 |  |  |  | 2 | 3 | 12 | контрольная работа/тестовый контроль |
| Тема 5. Библиотека STL | 14 | 3 |  |  | 4 | 3 | 4 | 12 |  |  |  |  |  | 12 | контрольная работа/тестовый контроль |
| **Раздел 6. ООП и написание GUI-программ** | | | | | | | | | | | | | | |  |
| Тема 1. Введение в визуальное Информатика и программирование | 15 | 3 |  |  | 4 | 3 | 5 | 12 |  |  |  |  |  | 12 | контрольная работа/тестовый контроль |
| Тема 2. Компоненты среды программирования. Формы | 15 | 3 |  |  | 4 | 3 | 5 | 12 |  |  |  |  |  | 12 | контрольная работа/тестовый контроль |
| Тема 3. Компонент Button | 14 | 3 |  |  | 4 | 2 | 5 | 12 |  |  |  |  |  | 12 | контрольная работа/тестовый контроль |
| Тема 4. Метка и текстовое поле | 15 | 4 |  |  | 4 | 2 | 5 | 13 |  |  |  |  |  | 13 | контрольная работа/тестовый контроль |
| Тема 5. Флажки и переключатели | 15 | 4 |  |  | 4 | 2 | 5 | 12 |  |  |  |  |  | 12 | контрольная работа/тестовый контроль |
| **Всего за семестр** | **140** | **32** |  |  | **36** | **27** | **45** | **144** | **6** |  |  | **8** | **9** | **121** |  |
| Форма контроля | экзамен  КСР – 4 часа | | | | | | | экзамен | | | | | | | Форма контроля |

## 4.2. Тематический план лекций

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № лекции | Тема занятия и вопросы лекции | Форма проведения (актив., интерактив.) | Количество часов | |
| ДФО | ЗФО |
| 1 | **Введение. Понятие программы, компилятора, исходного и исполняемого кода.**   1. Цели и задачи изучения дисциплины «Программирование» 2. Рекомендуемая литература по дисциплине 3. Понятие программы, компилятора и процесса компиляции 4. Понятие исходного и исполняемого кода 5. Понятие интегрированной среды программирования | С использованием мультимедиа | 2 | 1 |
| 2 | **Понятие алгоритма. Запись алгоритма на естественном и формальном языках. Графическая запись алгоритма**   1. Понятие алгоритма и его свойства 2. Формы записи алгоритма 3. Запись алгоритма на естественном языке 4. Запись алгоритма в графической форме 5. Запись алгоритма на псевдокоде | С использованием мультимедиа | 6 | 1 |
| 3 | **Императивная парадигма программирования.**  **Структура программы на языке С++.**   1. Понятие парадигмы программирования 2. Основные парадигмы программирования 3. Понятие императивной парадигмы программирования 4. Структура программы на языке С++ | С использованием мультимедиа | 2 | 2 |
| 4 | **Основы работы в среде программирования.**   1. Основные элементы рабочей области интегрированной 2. Понятие консольного приложения 3. Понятие проекта и решения в интегрированной среде 4. Создание и компиляция нового проекта | С использованием мультимедиа | 2 |  |
| 5 | **Типы данных в языке С++.**   1. Представление данных в компьютере 2. Целочисленный и вещественный тип 3. Символьный тип | С использованием мультимедиа | 2 | 2 |
| 6 | **Арифметические и логические выражения. Преобразование типов.**   1. Инструкция присваивания 2. Понятие синтаксиса и семантики 3. Арифметические выражений. Приоритет операций 4. Логические операции 5. Преобразование типов | С использованием мультимедиа | 2 |  |
| 7 | **Организация ввода/вывода в языке С++.**   1. Стандартная библиотека iostream 2. Стандартный ввод/вывод 3. Файловый ввод/вывод | С использованием мультимедиа | 4 |  |
| 8 | **Условный оператор и оператор выбора.**   1. Инструкция if 2. Вложенные инструкции if 3. Инструкция switch | С использованием мультимедиа | 2 |  |
| 9 | **Отладка и тестирование программ.**   1. Использование отладчика 2. Точки останова 3. Сценарий тестирования работы программы | С использованием мультимедиа | 2 |  |
| 10 | **Организация циклов в языке С++.**   1. Инструкция цикла с параметром 2. Инструкция цикла с предусловием 3. Инструкция цикла с постусловием | С использованием мультимедиа | 2 |  |
| 11 | **Побитовые операции**   1. Побитовые операции 2. Побитовые сдвиги 3. Применение побитовых операций | С использованием мультимедиа | 2 |  |
| 12 | **Функции.**   1. Описание функции 2. Формальные и фактические параметры 3. Значения параметров по умолчанию 4. Прототип функции | С использованием мультимедиа | 4 |  |
| 13 | **Понятие класса и объекта.**   1. Понятие класса и объекта. Понятие инкапсуляции 2. Атрибуты и методы 3. Объявление и определение класса 4. Доступ к членам класса | С использованием мультимедиа | 2 |  |
| 14 | **Конструктор и деструктор класса.**   1. Инициализация класса 2. Понятие конструктора и деструктора 3. Объекты классов | С использованием мультимедиа | 2 |  |
| 15 | **Пользовательские типы**   1. Структуры 2. Массивы | С использованием мультимедиа | 6 |  |
| 16 | **Указатели**   1. Понятие указателя 2. Объявление переменной типа указатель | С использованием мультимедиа | 4 | 2 |
| 17 | **Динамическое выделение памяти**   1. Выделение и высвобождение памяти 2. Ошибки при работе с памятью | С использованием мультимедиа | 4 |  |
| 18 | **Списки**   1. Структура список 2. Описание элемента списка 3. Основные операции со списком | С использованием мультимедиа | 4 |  |
| 19 | **Строки в С++**   1. Строка как массив символов   Библиотека string | С использованием мультимедиа | 2 |  |
| 20 | **Область видимости и время жизни**   1. Понятие области видимости 2. Локальные объекты 3. Глобальные объекты 4. Динамически размещаемые объекты | С использованием мультимедиа | 4 |  |
| 21 | **Функции: параметры ссылки и параметры-указатели**   1. Передача параметров по ссылке 2. Использование параметров-указателей | С использованием мультимедиа | 2 | 2 |
| 22 | **Раздельная компиляция. Заголовочные файлы**   1. Понятие раздельной компиляции 2. Написание заголовочного файла | С использованием мультимедиа | 2 |  |
| 23 | **Рекурсия и итерация**   1. Понятие рекурсии и итерации 2. Описание рекурсивной функции   Типовые задачи | С использованием мультимедиа | 4 |  |
| 24 | **Шаблоны функций**   1. Определение шаблона функции 2. Модели компиляции шаблонов | С использованием мультимедиа | 4 |  |
| 25 | **Перегруженные функции. Объявление перегруженных функций**   1. Понятие перегруженной функции и их использование 2. Объявление перегруженной функции 3. Перегрузка и область видимости   Указатели на перегруженные функции | С использованием мультимедиа | 4 | 2 |
| 26 | **Преобразование типов параметров перегруженных функций**   1. Точное соответствие 2. Повышение типов 3. Стандартное преобразование | С использованием мультимедиа | 2 |  |
| 27 | **Функции-кандидаты**   1. Функции-кандидаты 2. Подходящие функции | С использованием мультимедиа | 2 |  |
| 28 | **Перегрузка шаблонов функций**   1. Перегрузка шаблонов функций 2. Разрешение перегрузки при конкретизации | С использованием мультимедиа | 2 |  |
| 29 | **Перегруженные операторы**   1. Разработка перегруженных операторов 2. Оператор = 3. Оператор + 4. Оператор операторов ввода/вывода | С использованием мультимедиа | 4 |  |
| 30 | **Обработка исключений**   1. Возбуждение исключения 2. Перехват исключений | С использованием мультимедиа | 2 | 2 |
| 31 | **Директивы препроцессора**   1. Директива #define 2. Директивы условной компиляции 3. Директива #pragma | С использованием мультимедиа | 4 | 2 |
| 32 | **Библиотека STL**   1. Структура библиотеки 2. Контейнеры STL 3. Итераторы 4. Алгоритмы 5. Адаптеры | С использованием мультимедиа | 4 | 2 |
| 33 | **Введение в разработку GUI-программ**   1. Основные понятия 2. Технологии разработки GUI 3. Иерархия классов библиотеки QT 4. Пример простой GUI-программы | С использованием мультимедиа | 4 | 2 |
| 34 | **Элементы GUI**   1. Основные понятия 2. Кнопки 3. Обработка событий 4. Метки 5. Текстовое поле 6. Флажки и переключатели | С использованием мультимедиа | 10 | 2 |
|  | Итого: | | 96 | 22 |

## 4.3. Темы лабораторных занятий

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| №  занятия | Тема занятия | Форма проведения (актив., интерактив.) | | Количество часов | |
| ДФО | ЗФО |
|  | Представление алгоритмов в словесной форме, графической форме и в форме псевдокода | актив | | 6 | 2 |
|  | Первая программа на С++ | актив | | 2 | 2 |
|  | Создание проекта консольного приложения | актив | | 2 | 2 |
|  | Типы данных в языке С++ | актив | | 2 | 2 |
|  | Запись и выполнение арифметических и логических операций | актив | | 2 |  |
|  | Ввод/вывод. Escape-последовательности | актив | | 4 |  |
|  | Написание алгоритмов с ветвлением на языке С++ | актив | | 2 |  |
|  | Отладка и тестирование программы. Поиск семантических ошибок | актив | | 2 |  |
|  | Использование различных способов организации цикла | актив | | 6 |  |
|  | Побитовые операции | актив | | 2 |  |
|  | Определение функций и передача параметров | актив | | 4 | 2 |
|  | Описание класса | актив | | 2 |  |
|  | Написание конструктора и деструктора класса. | актив | | 2 |  |
|  | Описание структуры. Определение массива и операции над ним | актив | | 6 | 2 |
|  | Определение указателя и операции над ним | актив | | 4 | 2 |
|  | Работа с динамическим массивом | актив | | 4 | 2 |
|  | Односвязный список | актив | | 4 |  |
|  | Использование строковых переменных | актив | | 2 |  |
|  | Область видимости и время жизни | актив | | 4 |  |
|  | Параметры-ссылки и параметры-указатели | актив | | 2 | 2 |
|  | Написание пользовательского заголовочного файла | | актив | 2 |  |
|  | Рекурсия и итерация | | актив | 4 |  |
|  | Описание шаблона функции | | актив | 4 |  |
|  | Перегруженные функции | | актив | 4 | 2 |
|  | Типы параметров перегруженных функций | | актив | 2 | 2 |
|  | Функции-кандидаты | | актив | 2 |  |
|  | Перегрузка шаблонов функций | | актив | 2 |  |
|  | Перегруженные операторы | | актив | 6 |  |
|  | Обработка исключений | | актив | 2 |  |
|  | Директивы препроцессора | | актив | 4 |  |
|  | Библиотека STL | | актив | 6 |  |
|  | Разработка простого GUI-приложения | | актив | 4 | 2 |
|  | Разработка GUI-приложения «Калькулятор» | | актив | 12 |  |
|  | Итого: | |  | 106 | 24 |

# 5. Перечень учебно-методического обеспечения для самостоятельной работы обучающихся по дисциплине

1. Методические рекомендации по выполнению лабораторных работ по дисциплине «Информатика и программирование» для студентов очной и заочной форм обучения направления подготовки 09.03.03 «Прикладная информатика» / Ф.С. Ильясова, У.Б. Асанова. – РВУЗ «КИПУ», 2016-2017.
2. Прата С. Язык программирования С++: Лекции и упражнения. Учебник: Пер. с англ. / Стивен Прата. – СПб.: «ДиаСофтЮП», 2005. – 1104 с.
3. Шилдт Г. С++: руководство для начинающих / Г. Шилдт. – М.: Издательский дом «Вильямс», 2005. –672 с.
4. Страуструп Б. Язык программирования С++ / Б. Страуструп. – СПб.: Невский Диалект, 2000. - 991 с.
5. Кнут Дональд Э. Икусство программирования [Текст] = The art of computer programming: учебное пособие; Пер. с англ. / Дональд Э. Кнут; Под общ. ред. Ю.В.Козаченко. - 3-е изд., испр. и доп. - М., СПб., К. : Издат. дом "Вильямс", 2000 - Т. 1 : Основные алгоритмы. - 2000. - 720 с

## 5.1. Содержание самостоятельной работы студентов по дисциплине

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № | Наименование тем и вопросы, выносимые на самостоятельную работу | Форма отчетности | Количество часов | | Рекомендуемая литература |
| ДФО | ЗФО |
| 1 | *Тема 1. Введение в императивное программирование.*  Парадигмы программирования. Ознакомиться с базовыми парадигмами программирования: императивной, объектно-ориентированной, логической и функциональной. Изучить этапы развития и языки, реализующие эти парадигмы. Написать обзор языков программирования, представляющих базовые парадигмы | Контрольная работа | 24 | 60 | О-1, О-2, О-5 |
| 2 | *Тема 2. Элементы объектно-ориентированного программирования*  Ознакомиться с принципами объектно-ориентированного проектирования и объектно-ориентированного программирования. Написать обзор программного обеспечения для объектно-ориентированного проектирования. Продумать классы и их взаимодействие для реализации игры «Виселица» | Контрольная работа | 23 | 59 | О-1, О-2, О-3, О-4 |
| 3 | *Тема 3. Пользовательские структуры данных.*  Изучить различные алгоритмы сортировки (сортировка вставками, сортировка слиянием). Подготовить наглядный материал для демонстрации алгоритма сортировки. | Контрольная работа | 22 | 60 | О-1, О-2, О-4 |
| 4 | *Тема 4. Императивное программирование.*  Ознакомиться с особенностями описания и работы с двусвязным списком. Изучить особенности написания основных операций с двусвязными списками. Подготовить наглядный материал для демонстрации работы основных операций с двусвязными списками. | Контрольная работа | 22 | 60 | О-1, О-2, О-4 |
| 5 | *Тема 5. Перегруженные функции.*  Подготовить наглядный материал для демонстрации применения шаблонов функций | Контрольная работа | 22 | 60 | О-1, О-2, О-4 |
| 6 | *Тема 6. ООП и написание GUI-программ.*  Подготовить наглядный материал для демонстрации использования перегруженных функций или перегруженных операторов | Контрольная работа | 22 | 60 | Д-1 |
|  | Всего |  | 135 | 359 |  |

# 6. Фонд оценочных средств для проведения промежуточной аттестации обучающихся по дисциплине

## 6.1. Перечень компетенций с указанием этапов их формирования в процессе освоения образовательной программы

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Этапы формирования компетенции | Критерии сформированности на этапе | | | Оценочные средства |
| ПК-8 | ПК-2 | ПК-12 |
| Знаниевый этап (знать) | понятие алгоритма и свойств, различные записи формы алгоритма;  понятие двоичной системы исчисления и алгоритм перевода числа из десятичной системы в двоичную и обратно; | стандартные и пользовательские типы данных языка С++, основные понятия объектно-ориентированного программирования; синтаксис и семантику управляющих конструкций программы на языке С++. | понятие интегрированной среды программирования и её основные компоненты. | Тестовый контроль, теор. часть контрольной работы |
| Деятельностный этап (уметь) | выделять входные и выходные данные задачи, записывать алгоритмы в разной форме, выделять классы, их свойства и методы, | записывать и компилировать исходный код на языке С++;  использовать отладчик для поиска логических ошибок;  использовать компоненты библиотеки STL, | разделять исходный код на логически независимые составляющие; разрабатывать GUI. | Контрольная работа (практич.часть), лабораторные занятия, коллоквиум |
| Личностный этап (владеть) | навыками процедурного программирования, базовыми навыками объектно-ориентированного программирования | компиляцией многофайловых программ;  схемами хранения данных; | навыками обобщенного программирования (библиотекой STL), библиотекой Qt для разработки GUI приложений. | Экзамен |

## 6.2. Описание показателей и критериев оценивания компетенций на различных этапах их формирования, описание шкал оценивания

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Форма контроля** | **Уровни сформированности компетенции** | | | |
| Компетентность не сформирована | Пороговый уровень компетентности | Продвинутый уровень компетентности | Высокий уровень |
| Не удовл. | Удовл. | Хорошо | Отлично |
| Тестовый контроль | 1-59% правильных ответов | 60 -69%  правильных ответов | 70-89% правильных ответов | 90-100% правильных ответов |
| Контрольная работа | Выполнено правильно менее 30% теоретической части, практическая часть или не сделана или выполнена менее 30% | Выполнено не менее 50% теоретической части и практических заданий (или полностью сделано практическое задание) | Выполнено 51 -80% теор, части, практическое задание сделано полностью с несущественными замечаниями | Выполнено более 80% теоретической части, практическое задание выполнено без замечаний |
| Практическая работа (лабораторная работа) | Не выполнена или выполнена с грубыми нарушениями, выводы не соответствуют цели работы. | Выполнена частично или с нарушениями, выводы не соответствуют цели. | Работа выполнена полностью, отмечаются несущественные недостатки в оформлении. | Работа выполнена полностью, оформлена по требованиям. |
| Экзамен | Студент не знает значительной части теоретического материала по дисциплине, допускает существенные ошибки, неуверенно, с большими затруднениями выполняет практическое задание. | Студент имеет знания только основного материала, но не усвоил его деталей, допускает неточности, недостаточно правильные формулировки, нарушения логической последовательности в изложении программного материала, испытывает затруднения при выполнении практических работ. | Студент уверенно знает материал, грамотно и по существу излагает его, не допуская существенных неточностей в ответе на вопрос, правильно применяет теоретические положения при решении практических вопросов и задач, владеет необходимыми навыками и приемами их выполнения. | Студент глубоко и прочно усвоил программный материал, исчерпывающе, последовательно, четко и логически его излагает, умеет тесно увязывать теорию с практикой, свободно справляется с задачами, вопросами и другими видами применения знаний, причем не затрудняется с ответом при видоизменении заданий, использует в ответе материал монографической литературы, правильно обосновывает принятое решение, владеет разносторонними навыками и приемами выполнения практических задач. |

## 6.3. Типовые контрольные задания или иные материалы, необходимые для оценки знаний, умений, навыков и (или) опыта деятельности, характеризующих этапы формирования компетенций в процессе освоения образовательной программы

**Комплект тестов (тестовых заданий)**

по дисциплине «Информатика и программирование»

***ЧАСТЬ 1***

*В заданиях с 1 по 11 выберите один или несколько правильных вариантов ответа. В заданиях с 12 по 15 напишите расширенный ответ.*

**Тема 4.** Основы работы в среде программирования **(ПК-2)**

1. Определите значение суммы двоичных чисел 101+110

(а) 1111; (c) 0111;

(b) 1011; (d) 1010;

1. Если двоичное число 1011 перевести в десятичную систему счисления, то получится число …

(а) 11; (c) 13;

(b) 14; (d) 6;

1. Если десятичное число 33 перевести в двоичную систему счисления, то получится число …

(а) 100001; (c) 100010;

(b) 101001; (d) 100011;

1. Какие идентификаторы из ниже перечисленных недопустимы в С++?

(а) int double = 3.456;

(b) char 1\_or\_2 = ‘1’;

(c) int number = 3;

(d) double real\_num = 4.123;

1. Какие из приведенных ниже определений переменных содержат синтаксические ошибки?

(а) int car = 345, auto = 34.6;

(b) double mobile = 23.4;

(c) cin>>int tmp;

(d) chat alpha = ‘!’;

**Тема 6.** Арифметические и логические выражения **(ПК-8)**

1. Укажите правильные операторы присваивания

(а) int i; (c) int i;

i = 3.5; i ==3;

(b) int i; (d) int i;

i = 5; i = 3%5;

1. Укажите правильно записанные инструкции вывода:

(а) cout<<”This is the string and ”<<x<<”is the number”;

(b) cout<<”This is the string and ”x”is the number”;

(c) cin>>’This is the string and’>>x<<’is the number’;

(d) cout<<”This is the string and ”>>x<<”is the number”;

(e) cin<<”This is the string and ”<<x<<”is the number”;

(f) cout<<”This is the string and \n”<<x<<”\n is the number”;

**Тема 7**. Управляющие структуры языка С++ **(ПК-2)**

1. Укажите правильно записанные инструкции ввода:

(а) cout<<value;

(b) cout>> value;

(c) cin>> value;

(d) cout<< ‘value’;

(e) cin>> ‘value’;

(f) cout<<value>>;

**Тема 5**. Типы данных в языке С++ **(ПК-12)**

1. Какое значение будет иметь переменная val в результате выполнения данного фрагмента кода?

int val=3, i=7;

val=val\*i+(++i);

(а) 32;

(b) 28;

(c) 29;

1. Какое значение будет иметь переменная val в результате выполнения данного фрагмента кода?

int val=3, i=7;

val=val\*i+++i;

(а) 32;

(b) 28;

(c) 29;

1. Укажите определение переменной str, так чтобы приведенный ниже код не вызывал ошибок:

char str[…] = “My string ”;

str[9] = ‘1’;

cout<<str;

(а) char str[12] = “My string ”;

(b) char str[11] = “My string ”;

(c) char str[10] = “My string ”;

(d) char str[9] = “My string ”;

**Тема 1.** Понятие программы, компилятора, исходного и исполняемого кода **(ПК-8)**

1. Дайте определение понятия алгоритм. Перечислите и поясните свойства алгоритма.
2. Дайте определение понятия идентификатор. Перечислите правила его написания.
3. Как инициализация, так и присваивание обозначаются одним и тем же знаком «=». В чем отличие этих двух операций?
4. Как вы думаете, почему язык программирования назван С++, а не ++С?
5. Поясните понятие динамической типизации.

***ЧАСТЬ 2***

**Тема 2.** Понятие алгоритма. Запись алгоритма на естественном и формальном языках **(ПК-2)**

1. Напишите алгоритм в словесной форме и на псевдокоде для решения следующей задачи перехода пешехода через дорогу при отсутствии светофора.

**Тема 3.** Императивная парадигма программирования. Структура программы на языке С++ **(ПК-2)**

1. Заполните пропущенные фрагменты кода и при необходимости откорректируйте арифметическое выражение, так чтобы в результате переменная value принимала значение 2

*… value = 2,i;*

*… j = 2;*

*i = 5%value - sqrt(2\*j);*

*value = (value\*i+1)\*value+++i;*

## Оформление комплекта заданий для контрольной работы

ГБОУ ВО РК КИПУ

Кафедра \_\_прикладной информатики\_\_

(наименование кафедры)

**Комплект заданий для контрольной работы**

по дисциплине «Информатика и программирование»

**Контрольная работа №1**

**Вариант1**

**Теоретическая часть**

1. Понятие алгоритм. Свойства алгоритма. (ПК-12)

2. Описание функций. Формальные и фактические параметры. Значения параметров по умолчанию. (ПК-8)

**Практическая часть**

3. Задана последовательность символов, которая заканчивается символом '@'. Вычислить количество символов, которые являются заглавными буквами английского алфавита. (ПК-2, ПК-2, ОК-12)

4. Даны длины сторон катетов прямоугольного треугольника (*t1* и *t2*). Описать функцию, которая вычисляет и возвращает периметр треугольника. (ПК-8, ПК-2)

## 6.4. Методические материалы, определяющие процедуры оценивания знаний, умений, навыков и (или) опыта деятельности, характеризующих этапы формирования компетенций.

**Методические указания по выполнению**

Методические рекомендации по выполнению лабораторных работ по дисциплине «Информатика и программирование» для студентов очной и заочной форм обучения направления подготовки 09.03.03 «Прикладная информатика» / Ф.С. Ильясова, У.Б. Асанова. – РВУЗ «КИПУ», 2016-2017.

## 6.5. Итоговая рейтинговая оценка текущей и промежуточной аттестации студента по дисциплине

В ГБОУВО РК КИПУ используется рейтинговая 100-бальная система оценивания (см. Положение «О балльно-рейтинговой системе оценки знаний обучающихся Государственного бюджетного образовательного учреждения высшего образования Республики Крым «Крымский инженерно-педагогический университет»). Максимальное количество баллов по одной дисциплине за один семестр - 100 баллов независимо от количества зачетных единиц дисциплины. Максимальное количество баллов по дисциплине является суммой максимального количества баллов по всем контрольным мероприятиям дисциплины в текущем семестре.

За одно контрольное мероприятие (в том числе итоговое контрольное мероприятие) может быть начислено не более 40 баллов. Зачет по дисциплине выставляется по результатам текущего контроля.

По каждому контрольному мероприятию пороговое количество баллов, при котором контрольное мероприятие считается пройденным, составляет 40% от максимального количества баллов за данное контрольное мероприятие. Для необъективируемых контрольных мероприятий пороговое количество баллов устанавливается равным нулю.

Сумма баллов за необъективируемые контрольные мероприятия не может составлять более 30% от общей суммы баллов.

В случае если на момент проведения промежуточной аттестации будет установлено, что студент не принял участие в одном или более контрольных мероприятиях и (или) не набрал минимального количества баллов хотя бы по одному из них, студенту ставится отметка «неудовлетворительно».

Последовательность проведения контрольных мероприятий оформляется в виде графика контрольных мероприятий, который определяется УМК по дисциплине и утвержденным расписанием занятий на текущий семестр. Учебная программа дисциплины должна содержать не менее трех контрольных мероприятий за семестр. Конвертация баллов, набранных студентом по учебной дисциплине, в отметки осуществляется по следующей схеме:

* отлично» - от 81 до 100 баллов;
* «хорошо» - от 61 до 80 баллов включительно;
* «удовлетворительно» - от 41 до 60 баллов включительно;
* «неудовлетворительно» - от 0 до 40 баллов включительно;
* «зачтено» - от 41 баллов и выше;
* «не зачтено» - менее 41 баллов.

В зачетно-экзаменационную ведомость вносится оценка по четырехбалльной системе. По учебным дисциплинам, где итог оценивания уровня знаний студентов предусматривает экзамен, максимальная суммарная оценка текущего контроля (контрольных мероприятий) и промежуточного контроля должна составлять 100 баллов (например, 60 баллов текущего контроля (не менее трех контрольных мероприятий) и 40 баллов промежуточного контроля (экзамен)). По учебным дисциплинам, где итог оценивания уровня знаний студентов предусматривает зачет, максимальная суммарная оценка текущего контроля (контрольных мероприятий) должна составлять 100 баллов.

Зачет выставляется во время последнего семинарского (практического, лабораторного) занятия при условии, что суммарная оценка текущей аттестации студента превышает 41 балл («удовлетворительно» – и выше). Если студент набрал менее 41 балла, он сдает зачет на последнем семинарском (практическом, лабораторном) занятии.

Текущий контроль включает в себя проверку освоения студентом теоретических знаний и практических умений в ходе изучения учебного материала дисциплины в виде выполнения лабораторных работ (включая устный опрос при защите лабораторной работы), написания реферата (либо представления доклада) и др. Контрольная работа – это контроль уровня освоения студентом заочной формы обучения всего материала дисциплины в письменной форме.

***Шкала оценивания текущей и промежуточной аттестации студента***

|  |  |  |  |
| --- | --- | --- | --- |
| Уровни формирования компетенции | Сумма баллов по всем формам контроля | Оценка по четырехбалльной шкале | |
| для экзамена, курсового проекта (работы), практики | для зачета |
| Высокий | 90 – 100 | отлично | зачтено |
| Достаточный | 74-89 | хорошо |
| Базовый | 60-73 | удовлетворительно |
| Компетенция не сформирована | 0-59 | неудовлетворительно | не зачтено |

***Рейтинговая оценка текущего контроля за семестр для студентов ОФО***

|  |  |  |  |
| --- | --- | --- | --- |
| Форма контроля | Уровни формирования компетенций | | |
| Базовый | Достаточный | Высокий |
| 1 | 2 | 3 | 4 |
| Фронтальный опрос | 3-4 | 4-5 | 5-6 |
| Защита лабораторных работ | 7-9 | 9-11 | 11-12 |
| Модульный контроль | 10-12 | 12-13 | 13-14 |
| Домашние задания | 10-12 | 12-16 | 16-18 |
| Общая сумма баллов | **30-37** | **37-45** | **45- 50** |

***Рейтинговая оценка промежуточного контроля за семестр***

|  |  |  |  |
| --- | --- | --- | --- |
| Форма контроля | Уровни формирования компетенций | | |
| Базовый | Достаточный | Высокий |
| 1 | 2 | 3 | 4 |
| Теоретические вопросы | 13-15 | 15-18 | 18-20 |
| Практическое задание | 17-21 | 22-26 | 27-30 |
| Общая сумма баллов | **30-36** | **37-44** | **45-50** |

# 7. Перечень основной и дополнительной учебной литературы, необходимой для освоения дисциплины

**Основная литература**

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Библиографическое описание | Тип (учебник, учебное пособие, учебно-методическое пособие, практикум, др.) | Количество в библиотеке |
| 1 | Прата С. Язык программирования С++: Лекции и упражнения. Учебник: Пер. с англ. / Стивен Прата. – СПб.: «ДиаСофтЮП», 2005. – 1104 с. | Учебник | 5 |
| 3 | Фридман А. С/С++: Алгоритмы и приемы программирования / А. Фридман, Л. Кландер, Г. Шильдт. – М.: Бином, 2003. – 560 с.  Шилдт Г. С++: руководство для начинающих / Г. Шилдт. – М.: Издательский дом «Вильямс», 2005. –672 с. | учебное пособие | 5 |
| 4 | Страуструп Б. Язык программирования С++ / Б. Страуструп. – СПб.: Невский Диалект, 2000. - 991 с. | Книга | 5 |
| 5 | Кнут Дональд Э. Икусство программирования [Текст] = The art of computer programming : учебное пособие; Пер. с англ. / Дональд Э. Кнут; Под общ. ред. Ю.В.Козаченко. - 3-е изд., испр. и доп. - М., СПб., К. : Издат. дом "Вильямс", 2000 - Т. 1 : Основные алгоритмы. - 2000. - 720 с | Учебное пособие | 5 |

**Дополнительная литература**

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Библиографическое описание | Тип (учебник, учебное пособие, учебно-методическое пособие, практикум, др.) | Количество в библиотеке |
| 1 | Архангельский, А. Я. Информатика и программирование в C++ Builder 5 [Текст]: методический материал / А.Я. Архангельский. - М.: БИНОМ, 2000. - 1152 с. | Методический материал | 5 |
| 2 | Грегори, Кэйт. Использование Visual С++ 6. Специальное издание [Текст] : учеб. пособие / Кэйт Грегори; Под ред. Г.П. Петриковца. Пер. с анг. - М.; СПб.; К. : Издат. дом " Вильямс", 2000. - 864 с | Учебное пособие | 5 |
| 3 | Пол Айра. Объектно-Ориентированное Информатика и программирование на С++ [Текст] : монография / Айра Пол; Пер. с англ. Д.Ковальчука. - 2-е изд. - СПб. ; М.: Невский Диалект, 1999. - 462 с. - Библиогр.: с.445-446 . - Предм. указ.: с.447-461 . | Монография | 5 |
| 4 | Златопольский, Д. М. Я иду на урок информатики. Задачи по программированию. 7-11 кл. [Текст] : книга для учителя / Д. М. Златопольский. - М. : Первое сентября, 2001. - 208 с. | Книга | 5 |
| 5 | Кнут, Д. Искусство программирования. В 3 т. Т.1 [Текст] : основные алгоритмы: Пер. с англ. / Д. Кнут. - 3-е изд. - М. : Вильямс, 2001. - 720 с. | Книга | 5 |
| 6 | Телло Э. Объектно-ориентированное Информатика и программирование в среде Windows [Текст] : монография / Э. Телло; Пер.с англ. Д.М.Арапова, А.К.Петренко. - М. : Высш. шк., 1993. - 347 с. | Монография | 50 |

# 8. Перечень ресурсов информационно-телекоммуникационной сети Интернет, необходимых для освоения дисциплины

Лицензионные курсы, программы, ресурсы:

* <http://intuit.ru>, <http://ocw.mit.edu>, <https://www.coursera.org>, <http://www.udacity.com>
* <http://code.google.com/intl/>, <https://developer.mozilla.org/en-US/learn>
* <http://www.html5rocks.com/en/resources>, <http://thecodeplayer.com/>
* <http://www.codecademy.com/>, <http://www.khanacademy.org/>
* <http://generalassemb.ly/education/>, <https://peepcode.com/>
* <http://eloquentjavascript.net/>, <https://developer.apple.com/>
* <http://developer.android.com/index.html>

# 9. Методические указания для обучающихся по освоению дисциплины

## 9.1. Общие рекомендации по самостоятельной работе студентов

Подготовка современного бакалавра предполагает, что в стенах университета он овладеет методологией самообразования, самовоспитания, самосовершенствования. Это определяет важность активизации его самостоятельной работы. С целью организации данного вида учебных занятий необходимо в первую очередь использовать материал лекций и семинаров. Лекционный материал создает проблемный фон с обозначением ориентиров, наполнение которых содержанием производится студентами на семинарских занятиях после работы с учебными пособиями, монографиями и периодическими изданиями.

Самостоятельная работа формирует творческую активность студентов, представление о своих научных и социальных возможностях, способность вычленять главное, совершенствует приемы обобщенного мышления. Самостоятельная работа студентов по дисциплине предполагает более глубокую проработку ими отдельных тем курса, определенных программой. Основными видами и формами самостоятельной работы студентов поданной дисциплине являются: подготовка сообщений и докладов к практическим/семинарским занятиям; выполнение практических заданий; самоподготовка по вопросам; подготовка к дидактическому тесту, экзамену.

Важной частью самостоятельной работы является чтение учебной литературы. Основная функция учебников - ориентировать студента в системе тех знаний, умений и навыков, которые должны быть усвоены по данной дисциплине будущими специалистами. Учебник также служит путеводителем по многочисленным произведениям, ориентируя в именах авторов, специализирующихся на определённых научных направлениях, в названиях их основных трудов. Вторая функция учебника в том, что он очерчивает некий круг обязательных знаний по предмету, не претендуя на глубокое их раскрытие.

Чтение рекомендованной литературы – это та главная часть системы самостоятельной учебы студента, которая обеспечивает подлинное усвоение науки. Читать эту литературу нужно по принципу: «идея, теория, метод в одной,

в другой и т.д. книгах». Во всех случаях рекомендуется рассмотрение теоретических вопросов не менее чем по трем источникам. Изучение проблемы

по разным источникам – залог глубокого усвоения науки. Именно этот блок, наряду с выполнением практических заданий является ведущим в структуре самостоятельной работы студентов. Вниманию бакалавров предлагаются список литературы, вопросы к экзамену, а так же лабораторные задания.

Для успешного овладения курсом необходимо выполнять следующие

требования:

1) выполнять все домашние задания;

2) посещать занятия, т.к. весь тематический материал взаимосвязан между собой и, зачастую, самостоятельного теоретического овладения пропущенным материалом недостаточно для качественного его усвоения;

3) все рассматриваемые на занятиях вопросы обязательно фиксировать в отдельную тетрадь и сохранять её до окончания обучения в вузе;

4) проявлять активность при подготовке и на занятиях, т.к. конечный результат овладения содержанием дисциплины необходим, в первую очередь, самому бакалавру;

5) в случаях пропуска занятий по каким-либо причинам обязательно отрабатывать пропущенное преподавателю во время индивидуальных консультаций.

Внеурочная деятельность бакалавра по данной дисциплине предполагает:

- самостоятельный поиск ответов и необходимой информации по предложенным вопросам;

- выполнение заданий;

- выработку умений научной организации труда.

Успешная организация времени по усвоению данной дисциплины во многом зависит от наличия у бакалавра умения самоорганизовать себя и своё время для выполнения предложенных домашних заданий. Объём заданий рассчитан максимально на 2-3 часа в неделю. При этом алгоритм подготовки будет следующим:

1 этап – поиск в литературе теоретической информации по предложенным преподавателем вопросам;

2 этап – осмысление полученной информации, освоение терминов и понятий;

3 этап – составление плана ответа на каждый вопрос;

4 этап – поиск примеров по данной проблематике.

## 9.2 Методические рекомендации по освоению лекционного материала

Программой дисциплины предусмотрено чтение лекций в различных формах их проведения: проблемные лекции с элементами эвристической беседы, информационные лекции, лекции с опорным конспектированием, лекции-визуализации. На лекциях преподаватель рассматривает вопросы программы курса, составленной в соответствии с государственным образовательным стандартом. Из-за недостаточного количества аудиторных часов некоторые темы не удается осветить в полном объеме, поэтому преподаватель, по своему усмотрению, некоторые вопросы выносит на самостоятельную работу студентов, рекомендуя ту или иную литературу. Кроме этого, для лучшего освоения материала и систематизации знаний по дисциплине, необходимо постоянно разбирать материалы лекций по конспектами учебным пособиям. Во время самостоятельной проработки лекционного материала особое внимание следует уделять возникшим вопросам, непонятным терминам, спорным точкам зрения. Все такие моменты следует выделить или выписать отдельно для дальнейшего обсуждения на семинарском занятии. В случае необходимости обращаться к преподавателю за консультацией. Полный список литературы по дисциплине приведен в рабочей программе дисциплины.

## 9.3. Методические рекомендации по подготовке к лабораторным занятиям

Основной целью данного предмета является расширение научного кругозора и формирование практических навыков необходимых **академическому** бакалавру–инженеру-программисту. Подготовка студентов должна быть ориентирована на овладение специальностью инженера-программиста и развитии умений поиска наиболее эффективного решения задачи с последующей программной реализацией этого решения. ; формирование умения анализировать возникшую проблему, ставить на её основе исследовательские задачи и подбирать адекватный инструментарий для их решения; формирование стремления к постоянному самосовершенствованию, расширению палитры своего методического инструментария. В рамках ООП применяются лабораторные занятия, которые служат для закрепления усвоения теоретических навыков, полученных во время лекционных занятий. Каждая лабораторная работа оценивается от 1 до 10 баллов, в зависимости от сложности и затрачиваемого времени для выполнения лабораторной работы, качества выполненной работы студентом и т.д.

# 10. Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине (включая перечень программного обеспечения и информационных справочных систем (при необходимости))

* Методы обучения с использованием информационных технологий (демонстрация мультимедийных материалов, компьютерный лабораторный практикум и т.д.);
* Интернет-сервисы и электронные ресурсы (поисковые системы: Яндекс, Рамблер,Google;электронная почта: www.gmail.com- Почта gmail.com от Google).
* Программное обеспечение (Операционная система Windows, пакет прикладных офисных программ, программ для проведения анализа выборки данных).

# 11. Описание материально-технической базы, необходимой для осуществления образовательного процесса по дисциплине

Технические средства: персональные компьютеры, принтер, сканер, проектор, интерактивная доска;

# 12. Методические материалы к РПД

## Методические рекомендации по подготовке к лабораторным занятиям

**СЕМЕСТР 1**

**ПЛАН**

**ЛАБОРАТОРНОЙ РАБОТЫ № 1**

По дисциплине: Программирование

Тема занятия: Понятие алгоритма. Запись алгоритма

Цель занятия: научить разрабатывать алгоритм задачи и записывать его в разных формах.

Количество часов: 4

**Содержание работы (Задание, Задачи):** Написать алгоритм перехода через дорогу

1. Подойти к дороге.
2. Дождаться зелёного сигнала светофора.
3. Перейти дорогу.

Нет точного определения понятия алгоритм. Определяется оно интуитивно через свойства. Алгоритмом мы будем называть последовательность команд для исполнителя, которая должна быть:

- детерминированной (определенной) – каждый шаг должен быть точно определен, чтобы его мог выполнить любой исполнитель

- конечной, то есть заканчиваться через определенной число шагов

- массовой – то есть быть применимой к некоторому классу входных данных

Рассмотренный ранее алгоритм является линейным, то есть шаги выполняются один раз последовательно друг за другом.

Алгоритм может иметь ветвление, если для выполнения определенного шага требуется выполнение какого-либо условия.

**Задание 2.** Написать алгоритм перехода через дорогу при условии отсутствия светофора.

При необходимости многократного повторения ряда шагов алгоритм становится циклическим. Несколько шагов выполняется до тех пор, пока не выполнится определенное условие.

**Задание 3.** Усложнить задачу перехода через дорогу. Рассмотреть цикл с предусловием и постусловием.

Обратите внимание, что в первом случае цикл выполняется до тех пор, пока условие выполняется. Во втором случае цикл прекратит работу, как только выполнится поставленное условие.

**Задание 4.** В корзине n шаров. Посчитать количество желтых шаров в корзине.

**Задание 5.** Вы стоите перед магазином. Необходимо купить конфеты. (флажком может служить табличка открыто/закрыто)

**Пособия и инструменты:**

1. Microsoft Visual Studio 2010
2. Microsoft Visual Studio 2012
3. Microsoft Visual Studio Express

**Вопросы для защиты лабораторной работы:**

1. Понятие алгоритма
2. Свойства алгоритма
3. Формы записи алгоритмов

**ПЛАН**

**ЛАБОРАТОРНОЙ РАБОТЫ № 2**

По дисциплине: Программирование

Тема занятия: Словесное и формальное описание алгоритма

Цель занятия: научить разрабатывать алгоритм задачи и записывать его в разных формах.

Количество часов: 2

**Содержание работы (Задание, Задачи):**

**Задания низкого уровня сложности**

1. В корзине лежит несколько игрушек. Подсчитать количество всех плюшевых собак, рыжих котов и зеленых игрушек.
2. В ботаническом саду растут кактусы. Подсчитать количество цветущих кактусов и количество кактусов с желтыми цветами.
3. В академической группе учится n студентов. Подсчитать средний балл студентов, получивших на экзамене от 60 до 80 баллов.

**Задания высокого уровня сложности**

1. В корзине лежит несколько игрушек. Переложить в коробку всех плюшевых собак и рыжих котов.
2. В тарелке лежит три апельсина. Найти наибольший и наименьший из них.
3. В академической группе учится n студентов. Отметить в списке группы студентов, средний балл которых больше 4.

**Пособия и инструменты:**

1. Microsoft Visual Studio 2010
2. Microsoft Visual Studio 2012
3. Microsoft Visual Studio Express

**Вопросы для защиты лабораторной работы:**

1. Понятие псевдокода
2. Соглашения псевдокода

**ПЛАН**

**ЛАБОРАТОРНОЙ РАБОТЫ № 3**

По дисциплине: Программирование

Тема занятия: Знакомство со средой программирования

Цель занятия: научить создавать проекты, записывать код программы и компилировать в интегрированной среде прграммирования.

Количество часов: 2

**Содержание работы (Задание, Задачи):**

Интегрированная среда разработки (IDE – Integrated Development Environment) Microsoft Visual Studio 2008 позволяет разрабатывать программы на нескольких языках программирования: C++, C#, Visual Basic и ASP .NET. При первом старте вы можете увидеть окно, в котором среда попросит вас указать предпочитаемый вами язык программирования. От вашего выбора зависит, какие типы файлов и проектов будут предлагаться вам при создании нового приложения. Поскольку среда уже хотя бы раз была запущена, то язык уже был выбран ранее и теперь в окне среды будут отображаться соответствующие инструменты.

Итак, среда разработки состоит из следующих элементов:

* меню
* панель инструментов Стандартная
* область редактора
* окна инструментов в левой и нижней части окна.

Набор доступных инструментов зависит от типа создаваемого приложения.

В области окна редактора будет отображаться Начальная страница, состоящая из нескольких элементов:

* Последние проекты – ссылки на недавно созданные проекты и предложение создать новый или открыть существующий проект;
* Начало работы – ссылки на разделы файла помощи с информацией для тех, кто впервые работает с Visual Studio или впервые начинает разрабатывать программы;
* Новости от корпорации Microsoft.

Панели, расположенные слева и снизу от рабочей области являются настраиваемыми. В левой части окна интегрированной среды будет располагаться окно Обозревателя решений, а в нижней части окно выходных данных (в этом окне будет отображаться ход компиляции программы и найденные синтаксические ошибки).

Эти панели снабжаются заголовками, где отображается название панели и три кнопки:

меню настройки панели - эта кнопка вызывает меню настройки панели.

панель закреплена Visual Studio - если эта кнопка нажата, то панель будет закреплена на поверхности главного окна. Если копка отжата, то панель будет автоматически прятаться, когда она теряет фокус ввода.

закрытие панели Visual Studio - закрыть текущую панель. Ненужные панели, которые вы используете очень редко в своей работе можно закрыть.

Для отображения закрытой панели можно воспользоваться соответствующими командами меню Вид.

Поэкспериментируйте!

Вообще интегрированная среда предоставляет разработчику два типа контейнеров (структур) – проект (project) и решение (solution). Проект представляет собой набор файлов, объединённых для компиляции и сборки в одну программную единицу, такую как исполняемый модуль, динамически или статически подключаемая библиотека и т.д.

Все проекты, которые вы будете создавать в Visual Studio, заключаются в решение. Одно решение – это как папка для проектов и может содержать несколько проектов.

Visual Studio автоматически создает решение при создании нового проекта. При необходимости к решению можно добавлять другие проекты.

Итак создадим первый проект. Для этого:

1. В меню Файл выберите команду Создать –> Проект.

2. Среди предложенных типов проектов выберите Win32.

3. В окне шаблонов выберите Консольное приложение Win32. Консоль — это монитор и клавиатура, рассматриваемые как единое устройство. Консольное приложение — программа, предназначенная для работы в операционной системе MS-DOS (или в окне командной строки), для которой устройством ввода является клавиатура, а устройством вывода — монитор, работающий в режиме отображения символьной информации (буквы, цифры и специальные знаки). Однако такой тип приложений удобен на стадии обучения программированию.

4. В нижней части окна в строке Имя введите имя проекта (firstproject)

5. В поле Расположение нажмите на кнопку Обзор. В открывшемся окне перейдите в папку Рабочий стол и создайте в ней папку с именем И10. Убедитесь, что она выбрана и нажмите на кнопку Выбор папки. Теперь все файлы проекта будут сохраняться в вашей папке.

6. Нажмите ОК. В результате откроется окно мастера приложений Win32.

7. Здесь нажмите на кнопку Далее.

8. В появившемся окне Мастера убедитесь, что выбран тип Консольное приложение и установите флажок Пустой проект.

9. Нажмите Готово.

В результате будет создан новый проект, а в окне Обозревателя решений появится иерархия элементов решения, на верхнем уровне которой отображается имя вашего проекта. На следующем уровне иерархии находятся такие группы:

* Заголовочные файлы – файлы с информацией о функциях, используемых в данной программе.
* Файлы исходного кода – собственно файлы с текстом программы.
* Файлы ресурсов – например иконки и прочие графические объекты, используемые в данной программе.

Теперь создадим исходный файл программы, которая выводит на экран надпись «Hello Crimea!». Для этого:

1. Нажмите правой клавишей мыши изображении папки Файлы исходного кода в окне Обозревателя решений.
2. В появившемся меню выберите пункт Добавить–> Создать элемент.
3. В появившемся окне выделите шаблон Файл С++
4. В строке Имя введите имя файла исходного кода (source)
5. Нажмите Добавить

Файлы исходного кода программ, написанных на языке С++ имеют расширение cpp. В рабочей области появится окно редактора, в заголовке которого вы увидите имя нового исходного файла source.cpp. Обратите внимание, что ссылка на этот файл появилась в группе Файлы исходного кода в окне Обозревателя решений.

В окне редактора введите код программы для вывода на экран фразы «Hello Crimea!».

#include <iostream>

using namespace std;

void main()

{

cout<<”Hello world!”;

cin.get();

}

Теперь следует сохранить работу. Для этого на панели Стандартная (а также в меню Файл) имеется две команды: Сохранить и Сохранить все. При выборе первой будет сохранен лишь текущий cpp-файл. А при выборе второй команды будет сохранен весь проект.

Сохранить весь проект.

Теперь можно откомпилировать программу и запустить её на выполнение. Для этого на панели Стандартная нажмите на кнопку Начать отладку (либо выберите эту команду в меню Отладка, либо просто нажмите F5).

В окне Вывод будет выведена информация о ходе компиляции программы. Если компиляция проходит успешно, то появляется сообщение Компоновка, а затем происходит запуск программы. В результате на экране появится окно консольного приложения, в котором будет отображаться строка «Hello Crimea!». При нажатии на любую клавишу окно будет закрыто.

Теперь посмотрим, какие файлы были созданы в результате проделанной работы. Откройте в Проводнике папку И10. Здесь средой программирования была создана папка с именем проекта, то есть firstproject. В ней папки Debug и firstproject. В первой из них располагается исполняемый файл программы с именем firstproject.exe – то есть собственно разработанная вами программа. Попробуйте её запустить двойным щелчком мыши.

В папке firstproject находятся исходные коды. Здесь вы увидите файл source.cpp, то есть исходный код на С++, и файл проекта firstproject.vsproj. впоследствии именно файл проекта вы будете открывать в среде программирования с целью доработки программы.

**Пособия и инструменты:**

1. Microsoft Visual Studio 2010
2. Microsoft Visual Studio 2012
3. Microsoft Visual Studio Express

**Вопросы для защиты лабораторной работы:**

1. Понятие проекта
2. Содержимое папок
3. Порядок создания нового проекта

**ПЛАН**

**ЛАБОРАТОРНОЙ РАБОТЫ № 4**

По дисциплине: Программирование

Тема занятия: Типы данных. Арифметические и логические выражения

Цель занятия: научить записывать арифметические и логические выражения.

Количество часов: 2

**Содержание работы (Задание, Задачи):**

***Задание 1.***

Для демонстрации критической ситуации, когда результат вычислений не вмещается в участок памяти, выделенный в соответствие с выбранным типом, рассмотрим следующий пример.

Создайте новый проект, а в нем создайте исходный файл. Содержащий следующий текст:

#include<iostream>

using namespace std;

void main()

{

short x = 65530, y = 10;

x = x+y;

cout<<"65530+10="<<x;

cin.get();

}

Откомпилируйте. Какой результат вы получили? Почему?

Исправьте программный код так, чтобы результат получался правильный.

***Задание 2.***

По аналогии с предыдущими задачами напишите программу, которая выводит на экран размер переменной каждого типа (char, short, int, float, double) в байтах. Функция для вычисления размера sizeof(идентификатор). Причем вывод на экран должен быть типа:

*size of double = 8*

Здесь «size of double =» является строкой и выводится с помощью кавычек. А 8 является значением, которое возвращает функция sizeof.

**Пособия и инструменты:**

1. Microsoft Visual Studio 2010
2. Microsoft Visual Studio 2012
3. Microsoft Visual Studio Express

**Вопросы для защиты лабораторной работы:**

1. Стандартные типы данных и их размерность

**ПЛАН**

**ЛАБОРАТОРНОЙ РАБОТЫ № 5**

По дисциплине: Программирование

Тема занятия: Символьный тип. Ввод/вывод

Цель занятия: научить использовать символьный тип.

Количество часов: 2

**Содержание работы (Задание, Задачи):**

***Задание 1.***

На входе имеется строка st из 5 символов. Поменять местами первый и последний символы во входной строке. Результат вывести на экран.

*#include<iostream>*

*using namespace std;*

*void main()*

*{*

*char st[6];*

*char ch;*

*cin>>st;*

*ch=st[0];*

*st[0]=st[4];*

*st[4]=ch;*

*cout<<st;*

*cin.get();*

*cin.get();*

*}*

Дополнить программный код так, чтобы при запуске программы на экран выводился запрос «type characters and press Enter: ». Вывод строки символов после изменения первого и последнего символов должен предваряться строкой «the result is: »

***Задание 2.***

На входе два вещественных числа. Вывести сумму и разность чисел на экран в формате:

число 1 + число 2 = сумма

число 1 – число 2 = разность

Например,

2.567 + 4.389 = 6.8

Результирующие числа должны состоять не более, чем из 3 цифр.

**Пособия и инструменты:**

1. Microsoft Visual Studio 2010
2. Microsoft Visual Studio 2012
3. Microsoft Visual Studio Express

**Вопросы для защиты лабораторной работы:**

1. Понятие ASCII-таблицы
2. Чтение и вывод символов на экран
3. Форматированный вывод

**ПЛАН**

**ЛАБОРАТОРНОЙ РАБОТЫ № 6**

По дисциплине: Программирование

Тема занятия: Арифметические операции

Цель занятия: научить записывать арифметические выражения.

Количество часов: 2

**Содержание работы (Задание, Задачи):**

***Задание 1.***

На входе два вещественных числа num1 и num2. Написать программу для вычисления значения выражения:

1.1. .

1.2. 

1.3. 

1.4. 

1.5. 

1.6. 

При запуске программы на экране должна появиться строка-подсказка «enter number 1: », а после ввода первого числа – строка-подсказка «enter number 2: ». Вывод результата должен предваряться строкой «the result is: ». После вывода результата на экране должна быть выведена строка «press enter to exit», после которой при нажатии на клавишу Enter программа должна завершать работу.

***Задание 2.***

2.1. На входе задана строка из 5 символов. Поменять местами первый, третий и последний символы (первый заменить последним, последний третьим, третий первым). Вывод результата должен предваряться строкой «the result is: »

2.2. На входе заданы две строки по три символа. Вывести на экран символы, получающиеся сложением символов заданных строк. Вывод результата должен предваряться строкой «the result is: ».

**Пособия и инструменты:**

1. Microsoft Visual Studio 2010
2. Microsoft Visual Studio 2012
3. Microsoft Visual Studio Express

**Вопросы для защиты лабораторной работы:**

1. Арифметические операции
2. Приоритет операций

**ПЛАН**

**ЛАБОРАТОРНОЙ РАБОТЫ № 7**

По дисциплине: Программирование

Тема занятия: Преобразование типов

Цель занятия: научить использовать преобразование типов.

Количество часов: 2

**Содержание работы (Задание, Задачи):**

**Задание 1.**

Напишите программу, которая определяет и выводит на экран ASCII-код заданного символа. Используя эту программу, определите диапазон значений в ASCII-таблице, которые соответствуют заглавным буквам английского алфавита.

*#include<iostream>*

*using namespace std;*

*void main()*

*{*

*int code;*

*char ch;*

*cin>>ch;*

*code = (int) ch;*

*cout<<code;*

*cin.get();*

*cin.get();*

*}*

В строке *cin>>ch;* символ вводится с клавиатуры, после чего введенное значение приводится к типу int. В результате в переменной *code* будет содержаться ASCII-код введенного символа.

Поэкспериментируйте.

Дополнить программный код так, чтобы при запуске программы на экран выводился запрос «type characters and press Enter: ». Вывод результата должен предваряться строкой «ASCII-code is: ». В следующее строке должен выводиться символ предшествующий введенному. Для этого предварительно выведите на экран надпись «the previous character is: ». На следующей строке должен быть выведен символ, следующий за введенным. Для этого предварительно выведите на экран надпись «the next character is: ». Далее через одну строку должна выводится надпись «press Enter to exit».

*Например,*

*type characters and press Enter: A*

*ASCII-code is: 65*

*the previous character is: @*

*the next character is: B*

*press Enter to exit*

**Задание 2.**

Напишите программу для вычисления площади произвольного треугольника по заданным сторонам.

*Указание*: используйте формулу Герона , где *р* – полупериметр треугольника, *а*, *b* и *c* – его стороны

**Пособия и инструменты:**

1. Microsoft Visual Studio 2010
2. Microsoft Visual Studio 2012
3. Microsoft Visual Studio Express

**Вопросы для защиты лабораторной работы:**

1. Понятие преобразования типов
2. Явное и неявное преобразование

**ПЛАН**

**ЛАБОРАТОРНОЙ РАБОТЫ № 8**

По дисциплине: Программирование

Тема занятия: Инструкции ветвления и переключения

Цель занятия: научить использовать инструкции ветвления и переключения.

Количество часов: 2

**Содержание работы (Задание, Задачи):**

**Задание 1.** Запустите эту программу:

#include <iostream>

using namespace std;

void main()

{

setlocale(0, "");

double num;

cout << "Введите произвольное число: ";

cin >> num;

if (num < 10)

{

cout << "Это число меньше 10." << endl;

}

else

{

cout << "Это число больше либо равно 10." << endl;

}

}

**Поясните, что программа выполняет.**

Усовершенствуем программу так, чтобы она выводила сообщение, о том, что переменная num равна десяти:

if (num < 10)

{ cout << "Это число меньше 10." << endl;

}

else if (num == 10)

{

cout << "Это число равно 10." << endl;

}

else

{

cout << "Это число больше 10." << endl;

}

**Задание 2.**

Напишите программу для проверки принадлежности точки заданной области. Заполните пропущенные места в программном коде.

1

1

-1

-1

#include<iostream>

using namespace std;

void main()

{

… x, y;

cout<<”enter point coordinates:\n”;

cout<<”x=”;

cin>>x;

cout<<”y=”;

cin>>y;

if (…)

{

cout<<”this point belongs to area”;

}

else

{

cout<<”this point doesn’t belong to area”;

}

cin.get();

cin.get();

}

**Задание 3.**

Напишите программу-калькулятор, выполняющий сложение, вычитание, умножение и деление чисел.

#include<iostream>

using namespace std;

void main()

{

char sign;

double num1, num2, result;

cin>>num1;

cin>>sign;

cin>>num2;

switch (sign)

{

case '+':

{

result = num1 + num2;

break;

}

case '-':

{

result = num1 - num2;

break;

}

case '\*':

{

result = num1 \* num2;

break;

}

case '/':

{

result = num1 / num2;

break;

}

}

cout<<result;

cin.get();

cin.get();

}

Допишите программный код так, чтобы при запуске программы на экран выводился запрос ”enter number 1”. После ввода первого числа на экране должна появиться подсказка ”enter sign”. После ввода знака на экране должна появиться подсказка ”enter number 2”. Вывод результата должен предваряться строкой ”result=”.

Предусмотрите в программе проверку корректности знака арифметической операции. При вводе пользователем знака, отличного от '+', ‘-’, ‘\*’ и ‘/’ на экране должно появляться сообщение об ошибке ”error: wrong symbol”.

Предусмотрите в программе корректность ввода чисел для операции деления. В случае критической ситуации, когда делитель равен 0, программа должна вывести сообщение об ошибке ”error: division by zero”.

**Пособия и инструменты:**

1. Microsoft Visual Studio 2010
2. Microsoft Visual Studio 2012
3. Microsoft Visual Studio Express

**Вопросы для защиты лабораторной работы по исходному коду программы**

**ПЛАН**

**ЛАБОРАТОРНОЙ РАБОТЫ № 9**

По дисциплине: Программирование

Тема занятия: Инструкции организации цикла

Цель занятия: научить использовать инструкции организации цикла.

Количество часов: 4

**Содержание работы (Задание, Задачи):**

**Задание 1.**

На входе задана последовательность чисел *j*, которая заканчивается нулем. Вычислить значение выражения 

Предварительно напишите алгоритм решения задачи в словесной форме и на псевдокоде.

*Подсказка. Фрагмент кода для вычисления суммы*

*for(i=2; ; i+=2)*

*{*

*cout<<”enter j: ”;*

*cin>>j;*

*if (j!=0)*

*s = s +i\*j;*

*else*

*break;*

*}*

**Задание 2.**

Для заданного значения *t* и *n* вычислите значение многочлена 

*Решение задачи*

*#include<iostream>*

*#include<math.h>*

*using namespace std;*

*void main()*

*{*

*int n,i;*

*double t,a,f=0;*

*cout<<"enter n: ";*

*cin>>n;*

*cout<<"enter t: ";*

*cin>>t;*

*for(i=0; i<=n; i++)*

*{*

*cout<<"enter a: ";*

*cin>>a;*

*f = f + a\*pow(t,n-i);*

*}*

*cout<<"f = "<<f;*

*cin.get();*

*cin.get();*

*}*

Напишите решение это задачи без использования библиотеки математических функций.

**Пособия и инструменты:**

1. Microsoft Visual Studio 2010
2. Microsoft Visual Studio 2012
3. Microsoft Visual Studio Express

**Вопросы для защиты лабораторной работы по исходному коду программы**

**ПЛАН**

**ЛАБОРАТОРНОЙ РАБОТЫ № 10**

По дисциплине: Программирование

Тема занятия: Инструкции организации цикла

Цель занятия: научить использовать инструкции организации цикла.

Количество часов: 4

**Содержание работы (Задание, Задачи):**

**Задание 1.**

Задана последовательность символов, которая заканчивается символом «.». Вывести на экран ASCII-коды введенных символов.

Ниже приведен код программы, в котором пропущены фрагменты и допущены ошибки. Исправьте.

#include<iostream>

using namespace std;

void main()

{

char ch;

cout<<"enter the sequence of symbols\n";

do

{

cin>>ch;

cout<<ch<<'\n';

} while(ch...'.');

cout<<"finish";

cin.get();

cin.get();

}

Исправьте программный код так, чтобы было подсчитано количество введенных цифр, а код завершающегося символа («.») не выводился на экран

**Задание 2.**

Задана последовательность чисел. Вычислить сумму квадратов введенных чисел. Завершить программу, когда сумма превысит значение 100.

Ниже приведен код программы, в котором пропущены фрагменты и допущены ошибки. Исправьте.

#include<iostream>

using namespace std;

void main()

{

int val, s;

cout<<"enter the sequence of numbers\n";

while (s<=100)

{

...;

s=s+val\*val;

}

cout<<"s="<<s;

cin.get();

cin.get();

}

Исправьте код так, чтобы вычислялась сумма положительных чисел и квадратов отрицательных чисел.

**Задание 3.**

Задана последовательность чисел, которая заканчивается нулем. Вычислить сумму чисел больших 10 и меньших 25, произведение чисел больших 15 и вывести на экран наименьшее из введенных чисел.

Предварительно написать алгоритм решения задачи в словесной форме.

**Пособия и инструменты:**

1. Microsoft Visual Studio 2010
2. Microsoft Visual Studio 2012
3. Microsoft Visual Studio Express

**Вопросы для защиты лабораторной работы по исходному коду программы**

**ПЛАН**

**ЛАБОРАТОРНОЙ РАБОТЫ № 11**

По дисциплине: Программирование

Тема занятия: Инструкции ветвления и цикла

Цель занятия: закрепление пройденного материала.

Количество часов: 2

**Содержание работы (Задание, Задачи):**

**Задание 1.**

Напишите программу, которая выводит на экран числа от 1 до 5 следующим образом:

1 1 1 1 1

2 2 2 2

3 3 3

4 4

5

**Задание 2.**

Задано время в часах, минутах и секундах. Написать программу, которая переводит заданное время в секунды. Проверить корректность ввода времени, в случае некорректного ввода ждать следующих действий пользователя.

**Пособия и инструменты:**

1. Microsoft Visual Studio 2010
2. Microsoft Visual Studio 2012
3. Microsoft Visual Studio Express

**Вопросы для защиты лабораторной работы по исходному коду программы**

**ПЛАН**

**ЛАБОРАТОРНОЙ РАБОТЫ № 12**

По дисциплине: Программирование

Тема занятия: Отладка и тестирование

Цель занятия: научить использовать отладчик для поиска логических ошибок в коде.

Количество часов: 2

**Содержание работы (Задание, Задачи):**

Студенты получают cpp файл с исходным кодом решения задачи вычисления площади треугольника по заданным длинам сторон. Причем выполняется проверка корректности введенных длин (сумма двух любых длин должна быть больше третьей), выполняется проверка на равенство 0 введенных с клавиатуры длин. Если все длины нулевые то получается точка, о чем выводится соответствующее сообщение. Если длины заданы некорректно, то выводится сообщение о том, что это не треугольник. Во всех остальных случаях выводится сообщение о том, что это не геометрическая фигура.

В задаче допущены три ошибки, которые студенты должны исправить:

1. Тип переменной р (периметр) должен быть double, иначе значение периметра урезается до целого числа.
2. case 3 вместо case 4 поскольку задается всего три числа (три длины)
3. cout<<"\nsquare = "<<square; должно быть записано внутри case 1.

Срабатывание любой из этих ситуаций приводит к ошибке в работе программы.

Студенты должны воспользоваться отладчиком, найти ошибки и описать процесс поиска и устранение каждой из указанных ошибок. Описать в тетради.

Для проверки правильности работы программы можно использовать треугольник со сторонами 2, 3, 4. В результате должно получиться 2,904

**Пособия и инструменты:**

1. Microsoft Visual Studio 2010
2. Microsoft Visual Studio 2012
3. Microsoft Visual Studio Express

**Вопросы для защиты лабораторной работы по исходному коду программы**

**СЕМЕСТР 2**

**ПЛАН**

**ЛАБОРАТОРНОЙ РАБОТЫ № 1**

По дисциплине: Программирование

Тема занятия: Структурный тип

Цель занятия: научить определять пользовательский тип-структуру и использовать его для решения задач

Количество часов: 4

**Содержание работы (Задание, Задачи):**

***Задание 1.*** Заданы сведения о погоде: температура, осадки, направление ветра, скорость ветра. Ввод сведений заканчивается при вводе слова quit. Вычислить среднюю температуру и среднюю скорость ветра по заданным сведениям. Выяснить какой вид осадков встречается чаще всего.

Рассмотрим алгоритм решения задачи, записанный на псевдокоде. Для записи алгоритма на псевдокоде введем некоторые обозначения:

*temperature[x] –* значение поля температуры для текущего дня;

*speed[x] –* значение поля скорости ветра для текущего дня;

*precipitation[x] –* значение поля вида осадков для текущего дня.

1. *while (flag ≠”quit”)*
2. *do x ← GetWeather()*
3. *AvgT ← AvgT + temperature[x]*
4. *AvgS ← AvgS + speed[x]*
5. *if (precipitation[x] = “rain”)*
6. *then r ← r + 1*
7. *if (precipitation[x] = “snow”)*
8. *then s ← s + 1*

По заданному псевдокоду напишите программный код.

***Задание 2.*** В предыдущей задаче оформите ввод и вывод данных в следующем формате:

day 1:

temperature: …

wind: …

wind speed: …

precipitation: …

type quit to finish

day 2

…

Average temperature: …

Average wind speed: …

Precipitation: …

**Пособия и инструменты:**

1. Microsoft Visual Studio 2010
2. Microsoft Visual Studio 2012
3. Microsoft Visual Studio Express

**Вопросы для защиты лабораторной работы по исходному коду программы**

**ПЛАН**

**ЛАБОРАТОРНОЙ РАБОТЫ № 2**

По дисциплине: Программирование

Тема занятия: Массив

Цель занятия: научить создавать статический массив и производить наиболее типичные операции над его элементами.

Количество часов: 4

**Содержание работы (Задание, Задачи):**

***Задание 1.*** Заданы средние баллы n студентов. Упорядочить баллы в порядке убывания. Вычислить средний балл студентов группы. Вычислить количество студентов, балл которых превышает значение 4.

***Решение.***

Для решения этой задачи будем использовать статический массив. Разделим задачу на подзадачи и рассмотрим алгоритмы их решения.

Ниже представлен улучшенный алгоритм сортировки пузырьком. Этот алгоритм совершает n-1 шагов внешнего цикла, но меньше шагов внутреннего цикла алгоритма, учитывая число «всплывших элементов».Алгоритм записан на псевдокоде. Обратите внимание, что на псевдокоде, в отличие от С++, элементы массива нумеруются с 1. Это нужно учесть при написании кода и внести соответствующие поправки.

for j←1 to size do

for i←1 tosize-j do

if (st[i]> st [i+1])

then tmp ← st [i];

st [i] ← st [i+1];

st [i+1] ← tmp;

Для двух других подзадач будем определять функции Avg и BestScore соответственно. Первая функция должна возвращать вещественное число в качестве среднего балла, а вторая – целое число в качестве количества студентов. Обе функции в качестве параметра получают массив st.

Рассмотрим алгоритм решения подзадачи вычисления количества студентов, балл которых превышает значение 4.

for i←1 tosize do

if (st [i]≥4)

k ← k+1

***Задание 2.*** Модифицируйте код из первой задачи с учетом того, что теперь имеется не просто множество баллов студентов, а сведений о студентах. Сведения представляют собой балл и шифр студента. Необходимо вычисления из первой задачи. Однако теперь будем иметь дело не с массивом вещественных чисел, а с массивом структур-сведений о студентах.

Структура будет иметь такой вид:

struct student{

double score;

int id;

};

Следовательно массив сведений будет объявлен следующим образом:

student st[n];

*Примечание.* При выводе результатов сортировки достаточно вывести на экран шифры студентов.

**Пособия и инструменты:**

1. Microsoft Visual Studio 2010
2. Microsoft Visual Studio 2012
3. Microsoft Visual Studio Express

**Вопросы для защиты лабораторной работы по исходному коду программы**

**ПЛАН**

**ЛАБОРАТОРНОЙ РАБОТЫ № 3**

По дисциплине: Программирование

Тема занятия: Указатели

Цель занятия: выработать навыки использования указателей.

Количество часов: 4

**Содержание работы (Задание, Задачи):**

***Задание 1.*** Даны определения переменных.

int ival = 1024, ival2 = 2048;

int \*pi1 = &ival, \*pi2 = &ival2, \*\*pi3=0;

Что происходит при выполнении следующих операций присваивания? Допущены ли здесь ошибки?

1. ival = \*pi3;
2. \*pi2 = \*pi3;
3. ival = pi2;
4. pi2 = \*pi1;
5. pi1 = pi3;
6. ival = \*pi1;
7. pi1 = ival;
8. pi3 = &pi2;

***Задание 2.*** Данная программа содержит ошибку, связанную с неправильным использованием указателей. В чем состоит ошибка? Как можно её исправить?

int func(int \*pi)

{

\*pi = 1024;

return \*pi;

}

void main()

{

int \*pi2 = 0;

int ival = func(pi2);

return 0;

}

***Задание 3.*** Напишите функцию, которая вычисляет сумму элементов двумерного массива 4х4, расположенных в шахматном порядке.

**Пособия и инструменты:**

1. Microsoft Visual Studio 2010
2. Microsoft Visual Studio 2012
3. Microsoft Visual Studio Express

**Вопросы для защиты лабораторной работы по исходному коду программы**

**ПЛАН**

**ЛАБОРАТОРНОЙ РАБОТЫ № 4**

По дисциплине: Программирование

Тема занятия: Динамический массив

Цель занятия: научить создавать динамический массив и производить наиболее типичные операции над его элементами.

Количество часов: 6

**Содержание работы (Задание, Задачи):**

***Задание.*** Задано множество из n целых чисел. Число n произвольное.

1. Отсортируйте числа в порядке возрастания методом пузырька. Напишите соответствующую функцию.
2. Необходимо вычислить количество элементов множества, кратных трем. Напишите алгоритм, а затем реализуйте его программно в виде функции. Функция должна возвращать значение количества.
3. Напишите функцию, которая возвращает адрес минимального среди отрицательных элементов множества.
4. Напишите функцию поиска элемента множества с заданным значением. Воспользуемся бинарным поиском, который заключается в следующем. При поиске числа v в отсортированном массиве можно сначала сравнить искомый элемент со средним элементом массива и узнать, в какой половине его следует искать, а затем применить ту же идею.

Если элемент найден, то функция должна возвращать его адрес, в противном случае NULL.

Ниже представлен алгоритм поиска, записанный на псевдокоде:

BINARY\_SEARCH(A,v)

* 1. beg0
  2. endn
  3. k [(beg+end)/2]
  4. if A[k]=v then return k
  5. centre = k
  6. while (end-beg≠0)
  7. do k [(beg+end)/2]
  8. if A[k]=v then return k
  9. if (A[k]<v)
  10. then end k
  11. else beg  k
  12. return NULL

**Пособия и инструменты:**

1. Microsoft Visual Studio 2010
2. Microsoft Visual Studio 2012
3. Microsoft Visual Studio Express

**Вопросы для защиты лабораторной работы по исходному коду программы**

**ПЛАН**

**ЛАБОРАТОРНОЙ РАБОТЫ № 5**

По дисциплине: Программирование

Тема занятия: Списки

Цель занятия: научить описывать структуру «односвязный список» и производить над ним основные операции

Количество часов: 4

**Содержание работы (Задание, Задачи):**

***Задание.*** Задано множество целых чисел. Организуйте заданное множество в виде списка и напишите функции добавления нового элемента, поиска и удаления элемента с заданным ключом.

Ниже приведены определения функций, на основе которых необходимо описать заданные в задаче функции.

void Add(item\* L, int x)

{

item\* newitem = new item;

newitem->key = x;

newitem->next = L->next;

L->next = newitem;

}

item\* Delete(item\* L)

{

item\* deleted;

deleted = L->next;

L->next = L->next->next;

return deleted;

}

**Пособия и инструменты:**

1. Microsoft Visual Studio 2010
2. Microsoft Visual Studio 2012
3. Microsoft Visual Studio Express

**Вопросы для защиты лабораторной работы по исходному коду программы**

**ПЛАН**

**ЛАБОРАТОРНОЙ РАБОТЫ № 6**

По дисциплине: Программирование

Тема занятия: Строки

Цель занятия: научить использовать класс string

Количество часов: 2

***Задание.*** На входе дана строка символов. Подсчитать в ней количество согласных букв, а также каждой гласной буквы по-отдельности.

Ниже приведен код функции, которая устанавливает факт присутствия заданной буквы в строке. Используйте этот код как образец для написания функций, заданных в задаче.

void CheckLetter(string str, char letter)

{

int p = str.find(letter);

if (p != string::npos)

cout<<"letter was found on "<<p<<" position";

else

cout<<"letter was not found";

}

Здесь find(х) – метод класса string, который позволяет найти в строке заданный символ х.

**Пособия и инструменты:**

1. Microsoft Visual Studio 2010
2. Microsoft Visual Studio 2012
3. Microsoft Visual Studio Express

**Вопросы для защиты лабораторной работы по исходному коду программы**

**ПЛАН**

**ЛАБОРАТОРНОЙ РАБОТЫ № 7**

По дисциплине: Программирование

Тема занятия: Шаблоны функций

Цель занятия: научить описывать шаблоны функций

Количество часов: 4

**Содержание работы (Задание, Задачи):**

***Задание.***

Шаблон функции поиска максимального элемента массива задается следующим образом:

template<class T> T max (T \*x, int len)

{

T max = x [0];

for (int i=1; i<len; i++)

if (max<x [i])

max = x [i];

return max;

}

* + 1. По аналогии с этим шаблоном напишите шаблон функции поиска максимального элемента i-той строки двумерного массива.
    2. Используйте шаблон функции для поиска максимального элемента i-той строки двумерного массива целых чисел, символов и слов.

**Пособия и инструменты:**

1. Microsoft Visual Studio 2010
2. Microsoft Visual Studio 2012
3. Microsoft Visual Studio Express

**Вопросы для защиты лабораторной работы по исходному коду программы**

**СЕМЕСТР 3**

**ПЛАН**

**ЛАБОРАТОРНОЙ РАБОТЫ № 1**

По дисциплине: Программирование

Тема занятия: Структурный тип, указатели

Цель занятия: повторение основного материала предыдущего семестра

Количество часов: 2

**Содержание работы (Задание, Задачи):**

***Задание.*** Каждый студент формулирует условие задачи для своего сокурсника. Студенты обмениваются заданиями и выполняют их.

Условие задачи должно включать использование структурного типа, указателей и определение функций. Исходный код должен быть распределен как минимум на три файла: основной, файл с определениями функций. Заголовочный файл.

Объем задания должен быть рассчитан на 2 академических часа.

Преподаватель проверяет условия задач, разработанных студентами, прежде, чем они ими обменяются.

Защита лабораторной работы происходит в форме защиты отчета, который должен содержать словесный алгоритм решения задачи и листинг. Выполнение задачи преподаватель проверяет вместе с автором задачи.

Баллы выставляются студенту за выполнение каждой из двух ролей: автора задачи и исполнителя. С точки зрения автора задачи учитывается формулировка задания и умение критически оценить решение, найти сильные и слабые стороны предложенного решения. С точки зрения исполнителя задачи оценивается рациональность предложенного решения, аккуратность и грамотность оформления кода.

**Пособия и инструменты:**

1. Microsoft Visual Studio 2010
2. Microsoft Visual Studio 2012
3. Microsoft Visual Studio Express

**Вопросы для защиты лабораторной работы по исходному коду программы**

**ПЛАН**

**ЛАБОРАТОРНОЙ РАБОТЫ № 2**

По дисциплине: Программирование

Тема занятия: Перегрузка функций

Цель занятия: научить использовать перегруженные функции.

Количество часов: 4

**Содержание работы (Задание, Задачи):**

***Задание 1.*** Заданы два массива, упорядоченных по возрастанию. Написать функцию Merge, которая соединяет два заданных массива в один таким образом, чтобы он получился отсортированным по возрастанию. Напишите перегруженные функции для слияния массивов целых чисел, действительных, структур и слов.

Ниже приведен алгоритм слияния массивов А и В.

MERGE(A, B, p,q)

* 1. i<-0
  2. j<-0
  3. k<-0
  4. создать массив С длинной p+q
  5. while (i ≤ p and j ≤ q)
  6. do if A[i] < B[j]
  7. then C[k] <- A[i]
  8. i<-i+1
  9. else C[k] <- B[j]
  10. j<-j+1
  11. k<-k+1
  12. if i ≤ p
  13. then l<- i
  14. while k ≤ p+q
  15. do C[k] <-A[l]
  16. k<- k+1
  17. l<- l+1
  18. else l<- j
  19. while k ≤ p+q
  20. do C[k] <-В[l]
  21. k<- k+1
  22. l<- l+1

**Пособия и инструменты:**

1. Microsoft Visual Studio 2010
2. Microsoft Visual Studio 2012
3. Microsoft Visual Studio Express

**Вопросы для защиты лабораторной работы по исходному коду программы**

**ПЛАН**

**ЛАБОРАТОРНОЙ РАБОТЫ № 3**

По дисциплине: Программирование

Тема занятия: Векторы и деки библиотеки STL

Цель занятия: выработать навыки использования контейнеров библиотеки STL.

Количество часов: 4

**Содержание работы (Задание, Задачи):**

***Задание.*** Каждый студент формулирует условие задачи для своего сокурсника. Студенты обмениваются заданиями и выполняют их.

Условие задачи должно включать использование векторов или деков, методов этих классов. Исходный код должен быть распределен как минимум на три файла: основной, файл с определениями функций, заголовочный файл.

Объем задания должен быть рассчитан на 2 академических часа.

Преподаватель проверяет условия задач, разработанных студентами, прежде, чем они ими обменяются.

Защита лабораторной работы происходит в форме защиты отчета, который должен содержать словесный алгоритм решения задачи и листинг. Выполнение задачи преподаватель проверяет вместе с автором задачи.

Баллы выставляются студенту за выполнение каждой из двух ролей: автора задачи и исполнителя. С точки зрения автора задачи учитывается формулировка задания и умение критически оценить решение, найти сильные и слабые стороны предложенного решения. С точки зрения исполнителя задачи оценивается рациональность предложенного решения, аккуратность и грамотность оформления кода.

**Вопросы для защиты лабораторной работы по исходному коду программы**

**ПЛАН**

**ЛАБОРАТОРНОЙ РАБОТЫ № 4**

По дисциплине: Программирование

Тема занятия: Алгоритмы библиотеки STL

Цель занятия: выработать навыки использования контейнеров алгоритмов библиотеки STL.

Количество часов: 6

**Содержание работы (Задание, Задачи):**

***Задание.*** Заданы сведения о книгах: автор, название, издательство, год издания, количество страниц, шифр. Напишите программу, которая по команде пользователя будет выполнять одно из следующих действий:

1. Вводить сведения о новой книге.
2. Выводить в файл сведения о заданной книге (по шифру).
3. Сортировать множество книг по шифру.
4. Проверять и, если необходимо, редактировать записи так, чтобы автор и название книги были написано с большой буквы.
5. Выводить на экран сведения о первых n книгах.

В ходе решения задачи вам могут понадобиться следующие алгоритмы библиотеки STL:

find() - находит первое вхождение значения в последовательность

transform() - выполняет заданную операцию над каждым элементом последовательности

sort() - сортирует последовательность с хорошей средней эффективностью

stable\_sort() - сортирует последовательность, сохраняя порядок следования равных элементов

remove() - удаляет элементы с данным значением

count() - подсчитывает количество вхождений значения в последовательность

for\_earch() - выполняет операции для каждого элемента последовательности

**Пособия и инструменты:**

1. Microsoft Visual Studio 2010
2. Microsoft Visual Studio 2012
3. Microsoft Visual Studio Express

**Вопросы для защиты лабораторной работы по исходному коду программы**

**ПЛАН**

**ЛАБОРАТОРНОЙ РАБОТЫ № 5**

По дисциплине: Программирование

Тема занятия: первое знакомство с библиотекой Qt

Цель занятия: научить создавать GUI-приложение, ознакомиться с содержимым проекта программы.

Количество часов: 2

**Содержание работы (Задание, Задачи):**

Создайте новый проект. Для этого в меню Файл выберите Создать->Проект. В окне типов проектов выберите раздел Qt, а из предложенных шаблонов выберите Qt Application. Дайте имя проекту и сохраните его.

Автоматически будет создан целый ряд исходных и заголовочных файлов. Рассмотрим их назначение.

#include <QApplication>

#include <QLabel>

int main(int argc, char \*argv[])

{

QApplication app(argc, argv);

QLabel \*label = new QLabel("Hello Qt!");

label->show();

return app.execO;

}

В строках 1 и 2 в программу включаются определения классов QApplication и QLabel. Для каждого Qt-класса имеется заголовочный файл с тем же именем (с учетом регистра), содержащий определение этого класса.

В строке 5 создается объект QApplication для управления всеми ресурсами приложения. Для конструктора QApplication необходимо указывать параметры а где и argv, поскольку Qt сама обрабатывает некоторые из аргументов командной строки.

В строке 7 создается «виджет» текстовая метка QLabel, который выводит на экран сообщение «Hello Qt!» (здравствуй, Qt). По терминологии Qt и Unix виджетом (widget) называется любой визуальный элемент графического интерфейса пользователя. Этот термин происходит от «window gadget\* и соответствует элементу управления («control») и контейнеру («container») по терминологии Windows. Кнопки, меню, полосы прокрутки и фреймы являются примерами виджетов. Одни виджеты могут содержать в себе другие виджеты. Например, окно приложения обычно является виджетом, содержащим QMenuBar (панель меню), несколько QToolBar (панель инструментов), QStatusBar (строка состояния) и некоторые другие виджеты. Большинство приложений используют QMainWindow или QDialog в качестве окна приложения, однако Qt настолько гибка, что любой виджет может быть окном. В данном примере QLabel является окном приложения.

**Пособия и инструменты:**

1. Microsoft Visual Studio 2010
2. Microsoft Visual Studio 2012
3. Microsoft Visual Studio Express

**Вопросы для защиты лабораторной работы по исходному коду программы**

**ПЛАН**

**ЛАБОРАТОРНОЙ РАБОТЫ № 6**

По дисциплине: Программирование

Тема занятия: Использование компонентов кнопка и метка библиотеки Qt

Цель занятия: научить создавать простые GUI-программы

Количество часов: 4

**Содержание работы (Задание, Задачи):**Большинство приложений с графическим пользовательским интерфейсом име ют главное окно с панелью меню и инструментальной панелью, а также десятки ди алоговых окон, естественно дополняющих главное окно. Можно также создать при ложение из одного диалогового окна, которое будет непосредственно реагировать н выбор пользователя, выполняя соответствующие действия (например, таким прилс жением может быть калькулятор).

Рассмотрим пример диалогового окна Find. Нам понадобится два файла: finddialog. h и finddia-log. срр.

Сначала приведем файл finddialog. h:

#ifndef FINDDIAL0G\_H

#define FINDDIAL0G\_H

#include <QDialog.h>

class QCheckBox;

class QLabel;

class QLineEdit;

class QPushButton;

В строках с 4 по 7 даются предварительные объявления классов Qt, исполь­зующихся для реализации диалогового окна. Предварительное объявление (forward declaration) указывает компилятору С++ только на существование класса, не давая подробного определения этого класса (обычно определение класса содержится в его собственном заголовочном файле).

Затем мы определяем FindDialog как подкласс QDialog:

class FindDialog : public QDialog

{

Q\_OBJECT

public:

FindDialog(QWidget \*parent = 0);

signals:

void findNext(const QString &str, Qt::CaseSensitivity cs);

void findPrev(const QString &str, Qt::CaseSensitivity cs);

private slots:

void findClickedO;

void enableFindButton(const QString &text);

private:

QLabel \*label;

QLineEdit \*lineEdit;

QCheckBox \*caseCheckBox;

QCheckBox \*backwardCheckBox;

QPushButton \*findButton;

QPushButton \*closeButton;

};

#endif

Теперь рассмотрим файл f inddialog. срр, в котором находится реализация класса FindDialog.

#include <QtGui >

#include "finddialog.h"

FindDialog::FindDialog(QWidget \*parent)

QDialog(parent)

{

label = new QLabel(tr("Find &what:"));

lineEdit = new QLineEdit;

label->setBuddy(lineEdit);

caseCheckBox = new QCheckBox(tr("Match &case"));

backwardCheckBox = new QCheckBox(tr("Search backward"));

findButton = new QPushButton(tr("&Find"));

findButton->setDefault(true);

findButton->setEnabled(false);

closeButton = new QPushButton(tr("Close"));

connect(lineEdit, SIGNAL(textChanged(const QString &)),

this.SLOT(enableFindButton(const QString &)));

connect(findButton, SIGNAL(clickedO),

this..LOT(findClickedO));

connect(closeButton, SIGNAL(clickedO),

this.SLOT(closeO));

QHBoxLayout \*topLeftLayout = new QHBoxLayout;

topLeftLayout->addWidget(label);

topLeftLayout->addWidget(lineEdit);

QVBoxLayout \*leftLayout = new QVBoxLayout;

leftLayout->addLayout(topLeftLayout);

leftLayout->addWidget(caseCheckBox);

leftLayout->addWidget(backwardCheckBox);

QVBoxLayout \*rightLayout = new QVBoxLayout;

rightLayout->addWidget(findButton);

rightLayout->addwidget(closeButton);

rightLayout->addStretch();

QHBoxLayout \*mainLayout = new QHBoxLayout;

mainLayout->addLayout(leftLayout);

mainLayout->addLayout(rightLayout);

setLayout(mainLayout);

setWindowTitle(tr("Find"));

set FixedHeight(sizeHint().height());

}

Затем для размещения виджетов в окне мы используем менеджеры компоновки (layout managers). Менеджеры компоновки могут содержать как виджеты, так и другие менеджеры компоновки.

***Задание***. Изучите самостоятельно использование приложения Qt-дизайнер для разработки пользовательского интерфейса.

**Пособия и инструменты:**

1. Microsoft Visual Studio 2010
2. Microsoft Visual Studio 2012
3. Microsoft Visual Studio Express

## Вопросы для подготовки к экзамену по дисциплине

**Семестр 1**

1. Дайте определение понятия алгоритм. Перечислите и поясните свойства алгоритма.
2. Опишите структуру программы на языке С++.
3. Опишите императивную парадигму программирования. Приведите обзор языков.
4. Опишите объектно-ориентированную парадигму программирования. Приведите обзор языков.
5. Опишите функциональную парадигму программирования. Приведите обзор языков.
6. Опишите логическую парадигму программирования. Приведите обзор языков.
7. Целочисленные типы данных в языке С++.
8. Вещественные типы данных в языке С++.
9. Инструкция присваивания.
10. Приведение типов в языке С++.
11. Арифметические операции и их приоритеты.
12. Операции: инкремент и декремент.
13. Логические операции.
14. Стандартный ввод/вывод в языке С++.
15. Опишите инструкцию ветвления в языке С++.
16. Опишите инструкцию переключения в языке С++.
17. Опишите инструкцию организации цикла с параметром в языке С++.
18. Опишите инструкцию организации цикла с предусловием в языке С++.
19. Опишите инструкцию организации цикла с постусловием в языке С++.
20. Описание функций. Формальные и фактические параметры. Значения параметров по умолчанию.
21. Побитовые операции NOT, AND, OR и XOR
22. Побитовые сдвиги.
23. Понятие класса и объекта.
24. Понятие инкапсуляции и наследования.
25. Описание класса.
26. Инициализация класса. Конструктор и деструктор.

**Семестр 2**

1. Структурный тип.
2. Статический массив в С++. Основные операции. Использование в функциях
3. Тип указатель
4. Списки. Добавление элемента в односвязный список
5. Списки. Удаление элемента односвязного списка.
6. Списки. Поиск элемента с заданным ключом в односвязном списке
7. Динамическое выделение памяти
8. Класс String. Основные операции
9. Функции: параметры-ссылки
10. Функции: параметры-указатели
11. Локальная и глобальная область видимости. Время жизни локальных и глобальных переменных
12. Объявление и определение глобальных объектов. Сопоставление объявлений в разных файлах.
13. Определение пространства имен. Оператор разрешения имени
14. Using-объявления и using-директива
15. Раздельная компиляция
16. Создание и содержимое заголовочного файла. Пример
17. Рекурсивные функции
18. Объявление и определение шаблона функции
19. Конкретизация шаблона функции

**Семестр 3**

1. Определение перегруженных функций. Объявление перегруженных функций
2. Разрешение перегрузки
3. Преобразование типов параметров при разрешении перегрузки
4. Обработка исключений. Синтаксис и семантика блоков try… catch…
5. Понятие препроцессора. Директивы препроцессора
6. Директивы условной компиляции
7. Библиотека STL: её создание и назначение. Компоненты библиотеки STL
8. Векторы и их основные методы и свойства
9. Понятие итератора. Использование итераторов вектора
10. Деки и их основные методы и свойства
11. Списки и их основные методы и свойства
12. Адаптеры стек, очередь и очередь с приоритетами
13. Функциональные объекты библиотеки STL
14. Алгоритмы библиотеки STL
15. Библиотека Qt. Структура объектов библиотеки Qt
16. Сигналы и слоты
17. Назначение и особенности использование Qt-дизайнера
18. Свойства и методы компонентов Qt: кнопка, метка, текстовое поле

## Экзаменационные билеты

Первый вопрос экзаменационного билета проверяет сформированность компетенции ПК-2, второй – ПК-12, третий – ПК-8

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ № 1

1. Дайте определение понятия алгоритм. Перечислите и поясните свойства алгоритма.
2. Побитовые операции. Побитовые сдвиги
3. Задача.

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ № 2

1. Опишите структуру программы на языке С++.
2. Арифметические выражений. Преобразование типов.
3. Задача.

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ № 3

1. Опишите императивную парадигму программирования. Приведите обзор языков.
2. Описание функции. Формальные и фактические параметры.
3. Задача.

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ № 4

1. Опишите объектно-ориентированную парадигму программирования. Приведите обзор языков.
2. Арифметические и логические выражения.
3. Задача.

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ № 5

1. Опишите функциональную парадигму программирования. Приведите обзор языков.
2. Инструкция if.Вложенные инструкции if.
3. Задача.

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ № 6

1. Опишите логическую парадигму программирования. Приведите обзор языков.
2. Описание функций. Прототип функции.
3. Задача.

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ № 7

1. Целочисленные типы данных в языке С++.
2. Опишите инструкцию организации цикла с постусловием в языке С++.
3. Задача.

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ № 8

1. Вещественные типы данных в языке С++.
2. Опишите инструкцию организации цикла с предусловием в языке С++.
3. Задача.

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ № 9

1. Инструкция присваивания.
2. Опишите инструкцию организации цикла с параметром в языке С++.
3. Задача.

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ № 10

1. Приведение типов в языке С++.
2. Опишите инструкцию переключения в языке С++.
3. Задача.

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ № 11

1. Арифметические операции и их приоритеты.
2. Опишите инструкцию ветвления в языке С++.
3. Задача.

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ № 12

1. Операции: инкремент и декремент.
2. Стандартный ввод/вывод в языке С++.
3. Задача.

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ № 13

1. Логические операции
2. Дайте определение понятия алгоритм. Перечислите и поясните свойства алгоритма.
3. Задача.

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ № 14

1. Стандартный ввод/вывод в языке С++.
2. Опишите структуру программы на языке С++.
3. Задача.

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ № 15

1. Опишите инструкцию ветвления в языке С++.
2. Опишите императивную парадигму программирования. Приведите обзор языков.
3. Задача.

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ № 16

1. Опишите инструкцию переключения в языке С++.
2. Опишите объектно-ориентированную парадигму программирования. Приведите обзор языков.
3. Задача.

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ № 17

1. Опишите инструкцию организации цикла с параметром в языке С++.
2. Опишите функциональную парадигму программирования. Приведите обзор языков.
3. Задача.

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ № 18

1. Опишите инструкцию организации цикла с предусловием в языке С++.
2. Опишите логическую парадигму программирования. Приведите обзор языков.
3. Задача.

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ № 19

1. Опишите инструкцию организации цикла с постусловием в языке С++.
2. Целочисленные типы данных в языке С++.
3. Задача.

## Оформление комплекта заданий для контрольной работы

ГБОУ ВО РК КИПУ

Кафедра \_\_прикладной информатики\_\_

(наименование кафедры)

**Комплект заданий для контрольной работы**

по дисциплине «Информатика и программирование»

**РЕКОМЕНДАЦИИ К НАПИСАНИЮ И ПРАВИЛА ОФОРМЛЕНИЯ ТЕКСТА КОНТРОЛЬНОЙ РАБОТЫ**

**Студент выбирает номер своего варианта (выполняет один из предложенных уровней), согласно списку в журнале.**

**Структура текста**

Контрольная работа должна включать в себя: титульный лист, формулировку каждого задания и его решение, список использованных источников.

Один из критериев оценки – соответствие содержания ответа теме задания.

Каждый вариант состоит из следующих частей:

 *1-е и 2-е задания (теоретические вопросы)* – ответы на вопросы должны быть точными и полными, однако в них не должно быть ничего лишнего, не относящегося к теме. Также ответы должны результатом обработки лекционного материала, либо самостоятельной работы с материалом, но не копией сведений из источника.

 *3-е и 4-е задания (листинг программы, скриншоты)* – необходимо добавить в текст листинги с исходным кодом программы языке программирования C++ и скриншоты (Alt + PrtSc) выполнения программ.

**Список использованных источников** приводится по завершению работы. Фиксируются **только те** источники, с которыми фактически работал автор контрольной работы. В работе необходимо сделать ссылки на каждый используемый источник.

**Оформление текста контрольной работы**

Объем **не более 5–10 страниц** формата А4, шрифт – Times New Roman / Calibri, размер – **12 пт**, интервал – множитель 1.15 или 1.5. Поля страницы: **по 2 см**. Интервал между абзацами – **0 пт**.

При оформлении текста следует учитывать, что открывается работа титульным листом. Каждое задание начинается с новой страницы. Номера страниц ставятся внизу в середине листа. Титульный лист включается в общую нумерацию, но номер страницы на нем не ставится.

***Текст контрольной работы должен быть аккуратно оформлен; может быть дополнен иллюстративным материалом: схемами, таблицами, графиками.***

**Контрольная работа проверяться не будет в случае, если она не соответствует требованиям к оформлению.**

**ЗАДАНИЯ**

|  |  |  |
| --- | --- | --- |
| **Вариант** | **Содержание контрольной работы**  **Сложный уровень** | **Содержание контрольной работы**  **Базовый уровень** |
| 1 | 1. Понятие алгоритм. Свойства алгоритма.  2. Описание функций. Формальные и фактические параметры. Значения параметров по умолчанию.  3. Задана последовательность символов, которая заканчивается символом '@'. Вычислить количество символов, которые являются заглавными буквами английского алфавита.  4. Даны длины сторон катетов прямоугольного треугольника (*t1* и *t2*). Описать функцию, которая вычисляет и возвращает периметр треугольника. | 1. Понятие алгоритм. Свойства алгоритма.  2. Описание функций. Формальные и фактические параметры. Значения параметров по умолчанию.  3. Задана последовательность символов, которая заканчивается символом '@'. Вычислить количество символов в последовательности.  4. Даны длины сторон катетов прямоугольного треугольника (*t1* и *t2*). Вычислить периметр треугольника. |
| 2 | 1. Парадигмы программирования. Обзор языков.  2. Инструкция организации цикла с предусловием в языке С++.  3. Задана последовательность ASCII-кодов символов, которая заканчивается нулем. Вывести на экран символы, соответствующие наименьшему и наибольшему коду.  4. Даны три целых числа *a*, *b*, *c*. Описать функцию, которая вычисляет и возвращает их сумму (в функцию передаются все три числа). | 1. Парадигмы программирования. Обзор языков.  2. Инструкция организации цикла с предусловием в языке С++.  3. Задана последовательность из 10 символов. Вывести на экран ASCII-код каждого символа.  4. Даны значения трех целых чисел *a*, *b*, *c*. Если хотя бы одно из чисел равно нулю, вычислить сумму всех чисел. Иначе – вычислить их произведение. |
| 3 | 1. Происхождение языка программирования C++.  2. Инструкция организации цикла с параметром в языке С++.  3. Задана последовательность символов, которая заканчивается символом '#'. Вычислить количество символов, которые являются строчными буквами английского алфавита.  4. Даны длины сторон прямоугольника *a* и *b*. Описать функцию, которая вычисляет и возвращает периметр прямоугольника. | 1. Происхождение языка программирования C++.  2. Инструкция организации цикла с параметром в языке С++.  3. Задана последовательность символов, которая заканчивается символом '#'. Вычислить количество символов в заданной последовательности.  4. Даны длины сторон прямоугольника *a* и *b*. Вычислить периметр и площадь прямоугольника. |
| 4 | 1. Структура программы на языке C++.  2. Побитовые операции NOT, AND, OR и XOR и побитовые сдвиги.  3. Задана последовательность целых чисел, которая заканчивается нулем. Вычислить среднее арифметическое тех из них, которые принадлежат диапазону [5;15].  4. Дано целое число *x*. Описать функцию, которая определяет, является ли оно полным квадратом (пример: 9, 25, 64) и возвращает истину в этом случае, иначе – ложь. | 1. Структура программы на языке C++.  2. Побитовые операции NOT, AND, OR и XOR и побитовые сдвиги.  3. Даны значения двух целых чисел *x* и *y* (вводятся с клавиатуры). Используя цикл с параметром, вывести числа в диапазоне [*x*; *y*].  4. Дано целое число *x*. Если оно является полным квадратом (пример: 9, 25, 64), вывести на экран корень из данного числа. |
| 5 | 1. Компиляция и компоновка.  2. Инструкция организации цикла с постусловием в языке С++.  3. Задана последовательность целых чисел, которая заканчивается двумя нулями. Вычислить сумму отрицательных чисел последовательности.  4. Даны координаты двух точек (*x1*; *y1*) и (*x2*; *y2*). Описать функцию, которая вычисляет и возвращает длину отрезка. | 1. Компиляция и компоновка.  2. Инструкция организации цикла с постусловием в языке С++.  3. Задана последовательность целых чисел, которая заканчивается нулем. Вычислить произведение отрицательных чисел последовательности.  4. Даны координаты двух точек (*x1*; *y1*) и (*x2*; *y2*). Вычислить длину отрезка, образованного данными точками. |
| 6 | 1. Арифметические выражения. Преобразование и приведение типов.  2. Вещественные типы данных в языке C++.  3. Задана последовательность целых чисел, которая заканчивается двумя нулями. Вычислить произведение положительных чисел последовательности.  4. Дано целое число *j*. Описать функцию, которая возвращает его абсолютное значение |*j*| (без использования функций *abs()* и *fabs()*). | 1. Арифметические выражения. Преобразование и приведение типов.  2. Вещественные типы данных в языке C++.  3. Задана последовательность целых чисел, которая заканчивается нулем. Вычислить сумму положительных чисел последовательности.  4. Дано целое число *j*. Вычислить его абсолютное значение |*j*| (не используя функции *abs()* и *fabs()*). |
| 7 | 1. Операции отношений и их приоритеты. Логические операторы.  2. Компиляция и компоновка.  3. Каждая бактерия делится на две в течение минуты. В начальный момент времени имеется одна бактерия. Вычислите количество бактерий, которое появится через *n* минут.  4. Даны три целых числа *k, n, m*. Описать функцию, которая вычисляет и возвращает их произведение (в функцию передаются все три числа). | 1. Операции отношений и их приоритеты. Логические операторы.  2. Компиляция и компоновка.  3. Заданы целые числа *n* и *x*. Посчитать значение переменной *x* после *n* шагов, если с каждым шагом переменная *x* увеличивается вдвое.  4. Даны значения трех целых чисел *k, n, m*. Если все числа четные, вычислить их произведение. Иначе вычислить их сумму. |
| 8 | 1. Операции инкремента и декремента.  2. Парадигмы программирования. Обзор языков.  3. Задана последовательность символов, которая заканчивается символом '@'. Вычислить количество символов, встретившихся после первого символа '\*'.  4. Даны три целых числа *q*, *w*, *e*. Описать функцию, которая определяет и возвращает минимальное из них (в функцию передаются все три числа). | 1. Операции инкремента и декремента.  2. Парадигмы программирования. Обзор языков.  3. Задана последовательность из 10 целых чисел. Посчитать, сколько среди них чисел, которые по значению больше 18.  4. Даны значения трех целых чисел *q*, *w*, *e*. Определить минимальное из них. |
| 9 | 1. Арифметические операции и их приоритеты.  2. Инструкция switch.  3. Задана последовательность символов, которая заканчивается символом '@'. Вычислить количество символов, встретившихся до первого символа '! '.  4. Дано целое число *x*. Описать функцию, которая определяет, положительно ли оно и возвращает истину в этом случае, иначе – ложь. | 1. Арифметические операции и их приоритеты.  2. Инструкция switch.  3. Задана последовательность из 10 целых чисел. Посчитать среднее арифметическое только положительных чисел.  4. Дано целое число *x*. Если оно положительно, увеличить его в три раза. Иначе – увеличить на единицу. |
| 10 | 1. Целочисленные типы данных в языке C++.  2. Понятие алгоритм. Свойства алгоритма.  3. Задано целое число *n* и числа *a1, a2, …, an*. Вычислите сумму вида . *a1+2 a2+…+n an.*  4. Дано целое число *b*. Описать функцию, которая вычисляет и возвращает *b3*. | 1. Целочисленные типы данных в языке C++.  2. Понятие алгоритм. Свойства алгоритма.  3. Задано целое число *n*. Посчитать сумму чисел от 1 до *n*.  4. Даны целые числа *a* и *b*. Вычислить значение выражения *a4+ b3*. |
| 11 | 1. Вещественные типы данных в языке C++.  2. Структура программы на языке C++.  3. Задано целое число *n*. Вычислите значение суммы вида 1\*2+2\*3+3\*4+….+n\*(n+1)  4. Даны два целых числа *p* и *t*. Описать функцию, которая определяет, не равны ли они и возвращает истину в этом случае, иначе – ложь. | 1. Вещественные типы данных в языке C++.  2. Структура программы на языке C++.  3. Задано целое число *n*. Посчитать произведение только отрицательных чисел от 1 до *n*.  4. Даны значения двух целых чисел *p* и *t*. Если они равны, увеличить каждое число на 7. Иначе, вывести их произведение. |
| 12 | 1. Переменные. Правила именования. Оператор объявления и оператор присваивания.  2. Инструкция организации цикла с постусловием в языке С++.  3. На входе последовательность из *n* чисел. Вычислите среднее арифметическое четных чисел этой последовательности.  4. Даны три целых числа *a*, *b*, *c*. Описать функцию, которая определяет и возвращает максимальное из них (в функцию передаются все три числа). | 1. Переменные. Правила именования. Оператор объявления и оператор присваивания.  2. Инструкция организации цикла с постусловием в языке С++.  3. На входе последовательность из *n* чисел. Вычислите среднее арифметическое чисел этой последовательности.  4. Даны значения трех целых чисел *a*, *b* и *c*. Определить наибольшее из них. |
| 13 | 1. Структура программы на языке C++.  2. Операции отношений и их приоритеты. Логические операторы.  3. Задано целое число *n* и вещественное число *x*. Вычислить следующую сумму  4. Описать функцию, которая возвращает значение *true*, если произведение двух целых чисел *a* и *b* является положительным числом, и возвращает *false* в обратном случае. | 1. Структура программы на языке C++.  2. Операции отношений и их приоритеты. Логические операторы.  3. Задана последовательность из 10 целых чисел. Посчитать, сколько среди них чисел, которые по значению меньше 100.  4. Даны значения двух целых чисел *a* и *b*. Если оба числа положительные, записать в переменную *d* их среднее арифметическое. |
| 14 | 1. Стандартный ввод и вывод данных.  2. Описание функций. Формальные и фактические параметры. Значения параметров по умолчанию.  3. Задана последовательность целых чисел, которая заканчивается двумя нулями. Вычислить произведение чисел последовательности, расположенных в диапазоне [10;25].  4. Дано целое число *v*. Описать функцию, которая определяет, является ли оно нечетным и возвращает истину в этом случае, иначе – ложь. | 1. Стандартный ввод и вывод данных.  2. Описание функций. Формальные и фактические параметры. Значения параметров по умолчанию.  3. Задана последовательность из 8 целых чисел. Вычислить количество чисел последовательности, расположенных в диапазоне [10;25].  4. Дано целое число *v*. Определить, является ли оно нечетным. |
| 15 | 1. Парадигмы программирования. Обзор языков.  2. Инструкция if.  3. Задано целое число *n* и вещественное число *x*. Вычислить  следующую сумму  4. Описать функцию, которая возвращает значение *true*, если число *a* является средним геометрическим чисел *b* и *c*, и возвращает *false* в обратном случае. | 1. Парадигмы программирования. Обзор языков.  2. Инструкция if.  3. Задано целое число *n* и вещественное число *x*. Посчитать значение переменной *x* после *n* шагов, если с каждым шагом  переменная *x* увеличивается на 0.5.  4. Даны значения трех целых чисел *a*, *b* и *c*. Если число *a* является средним арифметическим чисел *b* и *c*, увеличить все три числа вдвое. Иначе, возвести их в квадрат. |
| 16 | 1. Инструкция switch.  2. Операции инкремента и декремента.  3. Вывести на экран таблицу стоимости 50, 100, 150, ..., 1000 г сыра (стоимость 1 кг сыра вводится с клавиатуры).  4. Даны два целых числа *a* и *b*. Описать функцию, которая определяет, равны ли они и возвращает истину в этом случае, иначе – ложь. | 1. Инструкция switch.  2. Операции инкремента и декремента.  3. Используя цикл с параметром вывести на экран последовательность: 50, 100, 150, …, 1000.  4. Даны значения двух целых чисел *a* и *b*. Если они не равны друг другу, вычислить их среднее арифметическое. |
| 17 | 1. Инструкция организации цикла с параметром в языке С++.  2. Операции отношений и их приоритеты. Логические операторы.  3. Вывести на экран таблицу стоимости 100, 200, ..., 2000 г конфет (стоимость 1 кг конфет вводится с клавиатуры).  4. Дано значение радиуса окружности *R*. Описать функцию, которая вычисляет и возвращает длину окружности. Описать функцию, которая вычисляет и возвращает площадь круга с данным радиусом. | 1. Инструкция организации цикла с параметром в языке С++.  2. Операции отношений и их приоритеты. Логические операторы.  3. Используя цикл с параметром вывести на экран последовательность: 100, 90, 80, …, 10.  4. Дано значение радиуса окружности *R*. Вычислить длину окружности и площадь круга с данным радиусом. |
| 18 | 1. Инструкция организации цикла с предусловием в языке С++.  2. Происхождение языка программирования C++.  3. Дана последовательность целых чисел, оканчивающаяся нулем. Посчитать, сколько в последовательности чисел, кратных 5 или 7.  4. Описать функцию, которая выводит на экран n строчек следующим образом:  *toy #1*  *toy #2*  *…*  *toy #n* | 1. Инструкция организации цикла с предусловием в языке С++.  2. Происхождение языка программирования C++.  3. Дана последовательность из 10 целых чисел. Посчитать, сколько в последовательности чисел, кратных 5.  4. Даны значения двух целых чисел *a* и *b*. Если они равны друг другу, вычислить их произведение и сумму. |
| 19 | 1. Понятие алгоритм. Свойства алгоритма.  2. Инструкция организации цикла с параметром в языке С++.  3. Запросите у пользователя 2 числа *x* и *y*. Посчитайте сумму чисел между заданными числами.  4. Дано целое число *v*. Описать функцию, которая определяет, является ли оно четным и возвращает истину в этом случае, иначе – ложь. | 1. Понятие алгоритм. Свойства алгоритма.  2. Инструкция организации цикла с параметром в языке С++.  3. Даны значения двух целых чисел *x* и *y*. Посчитать сумму чисел между заданными числами.  4. Дано целое число *v*. Если оно является четным, уменьшить его вдвое. |
| 20 | 1. Переменные. Правила именования. Оператор объявления и оператор присваивания.  2. Арифметические операции и их приоритеты.  3. Дана последовательность чисел, оканчивающаяся нулем. Найти *max* и *min* элементы и разницу между ними.  Описать функцию, которая 7 раз выводит фразу «New Year tree» на экран. | 1. Переменные. Правила именования. Оператор объявления и оператор присваивания.  2. Арифметические операции и их приоритеты.  3. Дана последовательность чисел, оканчивающаяся нулем. Найти минимальный элемент.  4. Даны значения двух целых чисел *a* и *b*. Вычислить их разницу по модулю (не используя функции *abs()* и *fabs()*). |
| 21 | 1. Арифметические выражения. Преобразование и приведение типов.  2. Инструкция if.  3. Вывести на экран таблицу квадратов нечетных чисел от 1 до *n* (*n* – задается с клавиатуры).  4. Описать функцию, которая выводит на экран n строчек следующим образом:  *1 – present*  *2 – present*  *…*  *n – present* | 1. Арифметические выражения. Преобразование и приведение типов.  2. Инструкция if.  3. Вывести на экран только нечетные числа от 1 до *n* (*n* – задается с клавиатуры).  4. Дано целое число *v*. Если, оно является нечетным, возвести его в куб. |
| 22 | 1. Происхождение языка программирования C++.  2. Описание функций. Формальные и фактические параметры.  Значения параметров по умолчанию.  3. Дана последовательность чисел, оканчивающаяся нулем. Найти *max* и *min* элемент, и определить расстояние между ними (разницу их позиций).  4. Дано целое число *x*. Описать функцию, которая определяет, меньше ли оно, чем число 2014 и возвращает истину в этом случае, иначе – ложь. | 1. Происхождение языка программирования C++.  2. Описание функций. Формальные и фактические параметры.  Значения параметров по умолчанию.  3. Дана последовательность чисел, оканчивающаяся нулем. Найти максимальный элемент.  4. Дано целое число *x*. Определить, меньше ли оно, чем число 2014. |
| 23 | 1. Побитовые операции NOT, AND, OR и XOR и побитовые сдвиги.  2. Стандартный ввод и вывод данных.  3. Задана последовательность из *n* чисел. Определить сколько раз меняется знак при переходе к следующему элементу.  4. Даны длины сторон прямоугольника (*a* и *b*). Описать функцию, которая вычисляет и возвращает площадь прямоугольника. | 1. Побитовые операции NOT, AND, OR и XOR и побитовые сдвиги.  2. Стандартный ввод и вывод данных.  3. Задана последовательность из *n* чисел. Определить сколько положительных и отрицательных чисел в последовательности.  4. Даны длины диагоналей ромба *a* и *b*. Вычислить площадь ромба. |
| 24 | 1. Инструкция if.  2. Целочисленные типы данных в языке C++.  3. Известно количество осадков, выпавших за каждый день двух недель. Определить среднедневное количество осадков за каждую неделю.  4. Даны длина стороны квадрата *d*. Описать функцию, которая вычисляет и возвращает длину диагонали. | 1. Инструкция if.  2. Целочисленные типы данных в языке C++.  3. Известно количество осадков, выпавших за каждый день недели. Определить среднедневное количество осадков за неделю.  4. Дана длина стороны квадрата *d*. Вычислить длину диагонали. |
| 25 | 1. Компиляция и компоновка.  2. Арифметические операции и их приоритеты.  3. Задано целое число *n* и вещественное число *x*. Вычислить следующую сумму  4. Описать функцию, которая возвращает значение *true*, если при известных величинах сторон *a*, *b* и *c*, треугольник является равносторонним, и возвращает *false* в обратном случае. | 1. Компиляция и компоновка.  2. Арифметические операции и их приоритеты.  3. Вывести на экран только четные числа от 1 до *n* (*n* – задается с клавиатуры).  4. Даны значения трех целых чисел *a*, *b* и *c*. Определить, является ли треугольник с данными сторонами равносторонним. |
|  | 1.Целочисленные типы данных в языке C++.  2. Переменные. Правила именования. Оператор объявления и оператор присваивания.  3. Запросите у пользователя 2 числа *x* и *y* (*x<y*). Посчитайте произведение отрицательных чисел между введенными числами.  4. Описать функцию, которая возвращает значение *true*, если при известных величинах сторон *a*, *b* и *c*, треугольник является равнобедренным, и возвращает *false* в обратном случае. | 1. Целочисленные типы данных в языке C++.  2. Переменные. Правила именования. Оператор объявления и оператор присваивания.  3. Запросите у пользователя 2 числа *x* и *y* (*x<y*). Выведите на экран числа в диапазоне [*x*, *y*].  4. Даны значения трех целых чисел *a*, *b* и *c*. Определить, является ли треугольник с данными сторонами равнобедренным. |
| 27 | 1. Вещественные типы данных в языке C++.  2. Стандартный ввод и вывод данных.  3. Задана последовательность из *n* целых чисел. Вычислите сумму чисел этой последовательности кратных 5.  4. Описать функцию, которая возвращает значение *true*, если сумма трех целых чисел *a*, *b* и *c* является числом, большим 1000, и возвращает *false* в обратном случае. | 1. Вещественные типы данных в языке C++.  2. Стандартный ввод и вывод данных.  3. Дана последовательность из 10 целых чисел. Посчитать, сколько в последовательности чисел, кратных 10.  4. Даны значения трех целых чисел *a*, *b* и *c*. Определить, является ли их сумма числом, большим 1000. |
| 28 | 1. Операции инкремента и декремента.  2. Инструкция организации цикла с постусловием в языке С++.  3. Задана последовательность символов, которая заканчивается символом '. '. Вычислить количество символов, введенных пользователем.  4. Описать функцию, которая возвращает значение *true*, если число *a* противоположно числу *b* (например, 5 и -5) и возвращает *false* в обратном случае. | 1. Операции инкремента и декремента.  2. Инструкция организации цикла с постусловием в языке С++.  3. Запросите у пользователя 2 числа *x* и *y* (*x>y*). Выведите на экран числа в диапазоне [*y*, *x*].  4. Даны значения двух целых чисел, a и b. Определить, являются ли они противоположными друг другу (например, 5 и -5). |
| 29 | 1. Арифметические выражения. Преобразование и приведение типов.  2. Инструкция организации цикла с предусловием в языке С++.  3. Задана последовательность целых чисел, которая заканчивается нулем. Посчитать, сколько в последовательности чисел, больших 100.  4. Даны длины всех сторон двух треугольников. Описать функцию, которая возвращает значение *true*, если периметры треугольников равны, и возвращает *false* в обратном случае. | 1. Арифметические выражения. Преобразование и приведение типов.  2. Инструкция организации цикла с предусловием в языке С++.  3. Задана последовательность из 10 целых чисел. Посчитать, сколько среди них чисел, которые по значению меньше 30.  4. Даны длины всех сторон двух треугольников. Вычислить периметр каждого треугольника. |
| 30 | 1. Побитовые операции NOT, AND, OR и XOR и побитовые сдвиги.  2. Инструкция switch.  3. На входе последовательность из *n* чисел. Вычислите среднее арифметическое положительных чисел этой последовательности.  4. Описать функцию, которая возвращает значение *true*, если число *a* является полным квадратом числа *b* (например, 25 и 5), и возвращает *false* в обратном случае. | 1. Побитовые операции NOT, AND, OR и XOR и побитовые сдвиги.  2. Инструкция switch.  3. На входе последовательность из *n* чисел. Вычислите количество положительных чисел этой последовательности.  4. Даны длины всех сторон двух прямоугольников. Вычислить площадь каждого прямоугольника. |
| 31 | 1. Вещественные типы данных в языке C++.  2. Инструкция организации цикла с параметром в языке С++.  3. Задана последовательность символов, которая заканчивается символом '@'. Посчитать сколько раз среди введенных символов встречаются символы '*d*' и '*t*'.  4. Описать функцию, которая возвращает значение *true*, если число *a* является средним арифметическим чисел *b* и *c*, и возвращает *false* в обратном случае. | 1. Вещественные типы данных в языке C++.  2. Инструкция организации цикла с параметром в языке С++.  3. Используя цикл с параметром вывести на экран последовательность: 2.1, 2.2, 2.3, …, 3.8.  4. Даны длины всех сторон двух прямоугольников. Вычислить диагональ каждого прямоугольника. |
| 32 | 1. Структура программы на языке C++.  2. Инструкция организации цикла с постусловием в языке С++.  3. Задана последовательность символов, которая заканчивается символом '@'. Вывести на экран ASCII-код каждого символа.  4. Описать функцию, которая возвращает значение *true*, если числа *a* и *b* равны друг другу и возвращает *false* в обратном случае. | 1. Структура программы на языке C++.  2. Инструкция организации цикла с постусловием в языке С++.  3. Используя цикл с параметром вывести на экран последовательность: 4.5, 4.6, 4.7, …, 6.2.  4. Даны значения двух целых чисел, a и b. Определить их разницу |
|  | *Примечания:*  *В последней задаче каждого варианта необходимо написать код всей программы, включая вызов функции.* | *Примечания:*  *В третьей задаче каждого варианта необходимо использовать циклы.*  *В четвертой задаче практически каждого варианта необходимо использовать условный оператор.* |
|  |  |  |

# Опорный конспект лекций

(по всем лекциям имеются презентации)