МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Дагестанский государственный университет»
Факультет информатики и информационных технологий

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ

Программирование на языке высокого уровня

Кафедра Информатики и Информационных технологий

Образовательная программа

09.03.02Информационные системы и технологии

Профиль подготовки:
Информационные системы и технологии

Уровень высшего образования:
бакалавр

Форма обучения
очная

Статус дисциплины:
базовая

Махачкала 2016

Рабочая программа по дисциплине «Программирование на языке высокого уровня» составлена в соответствии с требованиями Федерального государственного образовательного стандарта высшего образования по направлению подготовки 09.03.02 – Информационные системы и технологии от «12» марта 2015 г. № 219.

Составитель: Ахмедова З.Х, доцент каф. ИИиТ

Рабочая программа одобрена на заседании кафедры «Информатики и информационных технологии».

Протокол № 1 от 02.07 2016г

Зав кафедрой ИииТ _____ С.А. Ахмедов

Одобрена на заседании Методической комиссии факультета Информатики и информационных технологий

Протокол № 1 от 07.10 2016г

Председатель _____ Камилов К.Б.

Рабочая программа согласована с учебно-методическим управлением

7.10. 2016г _____

2

Аннотация.

Дисциплина«Программирование на языке высокого уровня » входит в вариативную по выбору часть образовательной программы бакалавриата по направлению09.03.02 Информационные системы и технологии.

Содержание дисциплины охватывает круг вопросов, связанных с программированием на языке С++. Очевидно, что применение объектно-ориентированного подхода делает программы понятнее, надежнее и проще в использовании.

Дисциплина нацелена на формирование следующих компетенций выпускника: профессиональных -ПК-33.

Преподавание дисциплины предусматривает проведение следующих видов учебных занятий: лекции, практические занятия, самостоятельная работа.

Рабочая программа дисциплины предусматривает проведение следующих видов контроля успеваемости в форме коллоквиум, устный опросипромежуточный контроль в форме зачета.

Объем дисциплины 2 зачетные единицы, в том числе в академических часах по видам учебных занятий

Семестр	Учебные занятия							Форма промежуточной аттестации (зачет, дифференцированный зачет, экзамен
	в том числе							
	Контактная работа обучающихся с преподавателем						СРС, в том числе экзамен	
	Всего	из них						
		Лекции	Лабораторные занятия	Практические занятия	КСР	консультации		
2	72	18	-	18	4		32	зачет

1.Цели задачи освоения дисциплины.

Целью изучения данной дисциплины является ознакомление студентов с языком программирования C++, в том числе средствами объектно-ориентированного программирования, а также освоение методикой построения объектно-ориентированных программ.

Каждая тема начинается с теоретического материала и содержит полное содержание необходимых понятий. В каждой теме даются задачи, которые необходимо выполнить на лабораторных занятиях с помощью программирования на языке C++. Теоретический материал, задачи и программы взаимосвязаны, и поэтому для полного усвоения обсуждаемых вопросов необходимо разобрать теоретический материал и решить все задачи. Лекционный материал построен так, что он доступен даже тем, кто изучает программирование «с нуля».

Учебный курс “Языки программирования высокого уровня ” рассчитан на один семестр и читается как дисциплина специализации на 1-ом курсе факультета информатики и информационных технологий. Курс состоит из 9 лекций (18 часов) и 9практичесикх занятий в компьютерном классе (18 часов).

2.Место дисциплины в структуре ООПВО.

Дисциплина Б1.В.ДВ.5 принадлежит вариативной по выбору части профессионального цикла дисциплинучебного плана по направлению подготовки “Информационные системы и технологии” и является одной из дисциплин, в рамках которой изучаютсяязыки и подходы и к программированию. Курс занимает важное место в профессиональнойподготовке специалиста по программированию. Он является одним из основныхобщепрофессиональных курсов, который лежит в основе изучения других предметов,связанных с программированием и алгоритмизацией. Знания, полученные в

результатепредмета также необходимы для выполнения курсовых и дипломных работ.

До изучения данного курса студентам необходимы знания в объеме школьного курсаинформатики и математикиЗнания, навыки и умения, приобретенные в результате прохождения курса, будут востребованы при изучении дисциплин специализаций, связанных с программированием такихкак численных алгоритмов, так и систем управления базами данных, а также в случае выполнения итоговой квалификационной работы, связанной с реализацией алгоритмов математического моделирования.

Чтение курса планируется в один семестр: начало курса во 2 семестре.

После изучения данной дисциплины студент должен **знать:**

1. Основные алгоритмические конструкции языка C++;
2. Основные средства библиотеки компилятора C++
3. Основные конструкции языков программирования C++
4. Средства объектно-ориентированного программирования на C++
5. Методику объектно-ориентированного анализа и проектирования.

После изучения данной дисциплины студент должен

уметь:

1. создавать программы на языке C++ с использованием среды разработки VisualStudio;
2. исправлять ошибки в программном коде;
3. выполнять отладку программ;
4. Написать программы на языках C++,в том числе, с использованием классов;
5. Работать с инструментальной системой программирования C++ Builder;

Обладать базовыми знаниями фундаментальных разделов информатики, а также иметь представление:

- о предмете, объектах и составных частях информатики;

- о математических, геометрических и физических основах информатики;
- об алгоритмизации, типах свойствах и способах представления алгоритмов;
- об областях применения прикладных программ;
- о компьютерных сетях и основах защиты информации.

Изучение данной дисциплины базируется на следующих дисциплинах:

1. математика
2. физика

Основные положения дисциплины должны быть использованы в дальнейшем при изучении следующих дисциплин:

1. Информационные технологии в химии
2. Моделирование информационных процессов и систем

3.Компетенции обучающего, формируемые в результате освоения дисциплины.

Процесс изучения дисциплины направлен на формирование элементов следующих компетенций в соответствии с ФГОС ВО по данному направлению:

Коды компетен	результаты освоения ООП *Содержание компетенций*	Перечень планируемых результатов обучения по дисциплине
ПК-33	способностью составлять инструкции по эксплуатации информационных систем	**Знать:** Концепцию императивного программирования, основные синтаксические конструкции языка, методы программирования **Уметь:** писать и отлаживать программы на языке С, осуществлять выбор и анализ соответствующих алгоритмов.

4. Объем, структура и содержание дисциплины.

4.1. Объем дисциплины составляет 2 зачетные единицы,

72академических часа.

4.2. Структура дисциплины.

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

№ п/п	Названия разделов	Семестр	Неделя	Виды учебной работы, включая самостоятельную работу студентов и трудоемкость (в часах)				Самостоятельная работа	Формы текущего контроля успеваемости (по неделям семестра) Форма промежуточной аттестации
				Лекции	Практические занятия	Лабораторные занятия	Контроль самост.работы		
1	2								
Модуль I. Языки и системы программирования.									
1	Введение в язык С++. Простые типы. Литералы. Операции. Типы данных и операции С++	2	1	2	2			8	Устный опрос
2	Базовые конструкции структурного программирования С++, организация ввода/вывода.	2	2	2	2			8	устный и письменный опросы
3	Составные типы. Циклы. Выражения отношений. Операторы ветвления.	2	3	2	2		2	6	проверка домашнего задания
	Итого за модуль:			6	6		2	22	
	Модуль II.Основные парадигмы программирования								
4	Функции, разновидности переменных; структура программы	2	4	2	2			6	к/р , тестовый контроль, устный и письменный опросы
5	Указатели и адресная арифметика	2	5	2	2			8	к/р , тестовый контроль, устный и письменный опросы
6	Модели памяти и пространства имен. Объекты и классы.	2	6	4	2			8	к/р , тестовый контроль, устный и письменный

									опросы
	Итого за модуль:			8	6			22	
	Модуль III.Основные парадигмы программирования								
7	Отношение «has-a».	2	7	2	2			10	к/р , тестовый контроль, устный и письменный опросы
8	Другие аспекты ООП. Обобщенное программирование в C++.	2	8	2	2		2	8	к/р , тестовый контроль, устный и письменный опросы
9	Работа с классами. Классы и динамическое распределение памяти	2	9		2			6	
	Итого за модуль:			4	6		2	24	
	Всего часов			**18**	**18**		**4**	**32**	**зачет**

4.3. Содержание дисциплины, структурированное по темам (разделам).

I –II модуль.

Тема 1. Основы языков C\C++:

- типы данных; операторы;
- выражения; строки; массивы;
- определяемые пользователем типы;
- указатели;

Тема 2. Функции.

- прототипы функции;
- ссылки;
- способы передачи параметров;

Тема 3. Классы и объекты.

- функции-элементы;
- дружественные функции;
- перегрузка функций и операторов;
- указатель this**;**

- наследование;
- виртуальные функции и полиморфизм;
- множественное наследование;
- абстрактные типы и классы;

Тема 4. Библиотека классов ввода-вывода C++.

- файловый ввод-вывод;

Тема 5. Шаблоны, исключения;

- функции-шаблоны;
- классы-шаблоны;

Тема 6. Создание приложения Windows.

- мастера; имена файлов и классов;
- шаблон документа;
- обработка сообщений;

Тема 7. Формирование ресурсов диалогового окна.

- создание класса модального диалогового окна;
- окна свойств и вкладки; принципы построения меню;
- динамические меню;
- многоуровневые меню;
- программирование вывода информации в приложении;

Тема 8. Расширение возможностей пользовательского интерфейса.

- введение в OLE;

Тема 9. Основы создания элементов управления ActiveX.

Темы практических занятий.

Тема: «Функции языка C++».
1. Имеется следующий образец структуры:

```
structbox
{
char maker[40];
```

float height;
float width;
floatlength;
floatvolume;
};

Создать функцию, в которую структура box передаётся по значению, а функция отображает значение каждого элемента структуры. Разработать программу, которая использует эту функцию. Элементы структуры создавать интерактивно.

2. Создать функцию Poisk _ w _ array (), которая принимает в качестве аргументов указатель на массив элементов данных типа int , искомое число x типа int и размер этого массива. Функция возвращает количество повторов числа x в массиве. Разработать программу, которая использует эту функцию. Элементы массива и число x создавать интерактивно (массив – динамический).

3. Создать функцию Preobr _ array (), которая принимает в качестве аргументов указатель на массив элементов данных типа double и размер этого массива. Функция увеличивает в два раза каждый элемент массива, а затем отображает содержимое массива. Разработать программу, которая использует эту функцию. Число элементов и сами элементы массива создавать интерактивно (массив – динамический).

4. Создать функцию Reverse _ array (), которая принимает в качестве аргументов указатель на массив элементов данных типа double и размер этого массива. Функция инвертирует порядок следования значений, хранимых в массиве, а затем отображает содержимое массива. Разработать программу, которая использует эту функцию. Число элементов и сами элементы массива создавать интерактивно (массив – динамический).

5. Создать функцию Ukazn _ cifra , которая принимает в качестве аргументов длинное натуральное число и искомую цифру, а возвращает количество повторов указанной цифры в записи числа. Разработать программу, которая использует эту функцию.

6. Создать функцию Reverse _ word , которая принимает в качестве аргумента указатель на слово (символьный массив), а возвращает указатель на новую строку, которая получается из исходной строки путём перестановки символов в обратном порядке. Разработать программу, которая использует эту функцию для проверки: является ли исходная строка «перевёртышем».

7. Создать функцию Preobr _ str (), которая принимает в качестве аргумента указатель на строку символов. Функция заменяет в исходной строке все точки и дефисы символом двоеточие и отображает полученную строку. Разработать программу, которая использует эту функцию (строку символов создавать динамически).

8. Создать функцию Preobr _ str (), которая принимает в качестве аргумента указатель на строку символов. Функция заменяет в исходной

строке сочетания символов “ rr ” на “ xx ” и отображает полученную строку. Разработать программу, которая использует эту функцию (строку символов создавать динамически).

9. Имеется следующий образец структуры:

```
structbox
{
char maker[40];
float height;
float width;
floatlength;
floatvolume;
};
```

Создать функцию, в которую передаётся адрес структуры box , а функция присваивает элементу структуры volume (объём) произведение остальных трёх измерений и отображает элементы структуры. Разработать программу, которая использует эту функцию. Элементы структуры создавать интерактивно.

10. Создать функцию Summa _ cifr , которая принимает в качестве аргумент а длинное натуральное число, а возвращает сумму цифр числа. Разработать программу, которая использует эту функцию.

11. Создать функцию Proizw _ cifr , которая принимает в качестве аргумент а длинное натуральное число, а возвращает произведение цифр числа. Разработать программу, которая использует эту функцию.

12. Создать функцию Right _ cifri , которая принимает в качестве аргументов длинное натуральное число и заданное количество цифр n , а возвращает число, полученное из «правых» n цифр исходного числа. Разработать программу, которая использует эту функцию.

13. Создать функцию Leftt _ cifri , которая принимает в качестве аргументов длинное натуральное число и заданное количество цифр n , а возвращает число, полученное из «левых» n цифр исходного числа. Разработать программу, которая использует эту функцию.

14. Создать функцию Sum _ deliteli , которая принимает в качестве аргумента натуральное число, а возвращает сумму делителей исходного числа. Разработать программу, которая использует эту функцию.

15. Создать функцию Right _ simb (), которая принимает в качестве аргументов указатель на строку символов и натуральное число n . Функция создаёт новую строку из n правых символов исходной строки и отображает полученную строку. Разработать программу, которая использует эту функцию (строку символов создавать динамически).

16. Создать функцию Left _ simb (), которая принимает в качестве аргументов указатель на строку символов и натуральное число n . Функция создаёт новую строку из n левых символов исходной строки и отображает полученную строку. Разработать программу, которая использует эту функцию (строку символов создавать динамически).

Тема: «Классы. Заимствование функциональности одного класса другим. Решение задач планиметрии».

1. Дано множество точек на плоскости. Указать в нём две такие точки, чтобы отрезок, концами которого они являются, был параллелен оси Ох.

2. Дано множество точек на плоскости. Указать в нём две такие точки, чтобы отрезок, концами которого они являются, был параллелен оси Оу.

3. Дано множество точек на плоскости. Указать в нём две такие точки, чтобы прямая, проходящая через них, делила 1 и 3 координатные углы пополам.

4. Дано множество точек на плоскости. Указать в нём две такие точки, чтобы прямая, проходящая через них, делила 2 и 4 координатные углы пополам.

5. Дано множество точек на плоскости. Выяснить, существует ли такая точка в данном множестве, что все остальные точки этого множества лежат на окружности с центром в этой точке.

6. Дано множество точек на плоскости. Сколько точек данного множества попадает в круг с центром в начале координат и радиусом 5 ед.?

7. Дано множество точек на плоскости. Сколько точек этого множества окажутся вне круга, ограниченного окружностью с центром в последней точке этого множества и радиусом, равным 2?

8. Дано множество точек на плоскости. Найдите такую окружность с центром в начале координат, накоторой лежит наибольшее количество точек данного множества.

9. Дано множество точек на плоскости. Сколько отрезков можно построить на основе этого множества точек, чтобы они были параллельны оси Ох?

10. Дано множество точек на плоскости. Сколько отрезков можно построить на основе этого множества точек, чтобы они были параллельны оси Оу?

11. Дано множество точек на плоскости. Найдите пару точек – центр и точка на окружности – так, чтобы круг, ограниченный этой окружностью, содержал все остальные точки данного множества.

12. Дано множество точек на плоскости. Указать в нём две такие точки, чтобы все остальные точки находились по одну сторону от прямой, проходящей через две указанные точки.

13. Дано множество точек на плоскости. Выяснить, лежат ли эти точки на одной прямой.

14. Дано множество точек на плоскости. Найти две точки так, чтобы у первой Y-вая координата была самой маленькой, у второй Y-вая координата была самой большой и вычислить расстояние между этими точками.

15. Дано множество точек на плоскости. Можно ли в данном множестве найти пару точек X и Y так, чтобы вне круга, для которого отрезок XY является диаметром, находились все остальные точки данного множества?

16. Дано множество точек на плоскости и прямая d своим уравнением

X – 2 Y + 5 = 0. По разные стороны от этой прямой найти 2 самые близкие точки из данного множества.

17. Дано множество точек на плоскости. Выбрать, если возможно, в этом множестве 2 точки так, чтобы из оставшихся точек ни одна больше не принадлежала прямой, проходящей через выбранные 2 точки.

Примерный перечень вопросов к зачету (2 семестр)**.**

1. Основы языков C / C ++: синтаксис языка, директивы препроцессора, консольное приложение.
2. Коллизия имён и использование пространства имен.
3. Классификация типов данных в C++. Арифметические типы данных. Преобразование типов в выражениях.
4. Переменные, константы, операторы и выражения. Приоритеты операций и операторов в C ++.
5. Потоки и файлы; ввод, вывод, управляющие последовательности. Библиотечные функции.
6. Функции: прототип, определение, реализация. Функция main (). Структура C++-программы.
7. Логические операции. Операторы управления C++-программой.
8. Составной тип – строки. Методы ввода строк.
9. Составные типы – структуры, объединения, перечисления и определяемые пользователем типы.
10. Виды циклов в C++.
11. Указатели и свободная память. Работа с памятью с помощью new и delete .
12. Указатели и строки.
13. Массивы. Одномерные и многомерные массивы. Динамические массивы.
14. Статические и динамические массивы структур.
15. Ссылки. Передача массивов как параметров.
16. Функции пользователя в C++. Передача параметров в функции по значению и по ссылке.
17. Функции и строки. Функции сравнения, копирования строк.
18. Функции и структуры. Передача параметров (типа «структура») в функции.
19. Объекты и классы. Описание класса.

20. Конструкторы, их виды. Деструкторы.

21. Inline -функции, создание inline -функций внутри класса.

22. Статические и динамические классы.

23. Область видимости и классы памяти (auto , register , extern , static).

24. Указатели на объекты. Передача объектов функциям. Объекты в качестве возвращаемых функциями значений.

25. Массивы объектов, организация работы с ними.

26. Полиморфизм. Переопределение функций. Функции с аргументами по умолчанию.

27. Конструктор копий.

28. Виртуальные функции. Абстрактные классы. Раннее и позднее связывание.

29. Указатель this .

30. Традиционные типы конструкторов. Списки инициализаторов.

31. Перегрузка функций. Неоднозначность при перегрузке функций.

32. Перегрузка операторов.

33. Реализация операторной функции =.

34. Реализация операторной функции сравнения ==.

35. Наследование. Виды наследования. Дружественные функции.

36. Перегрузка операторов ввода/вывода.

37. Вывод в файл.

38. Ввод в файл.

39. Шаблоны функций, пример.

40. Шаблоны классов, пример.

41. Обработка исключительных ситуаций.

42. Обзор С ++: объектно-ориентированное программирование - инкапсуляция, полиморфизм, наследование.

5.Образовательные технологии.

В учебном процессе помимо традиционных форм проведения занятий используются лекции – визуализации, лекции – диалоги.Лабораторные занятия проводятся в компьютерном классе с использованиемИнтернет среды.При проведение практических занятий используются деловые игры с разбором конкретных ситуаций.

- Лекционные занятия
- Традиционные технологии
- Иллюстрация работы алгоритмов с использованием видео и элементов анимации в презентациях.

- Демонстрация элементов современных методов разработки программ с использованием видеопроектора
- Практические занятия
- Традиционные технологии
- Коллективное выполнение заданий с использованием видеопроектора, среды разработчика и системы контроля версий исходного кода SVN или Git
- Лабораторные занятия
- Традиционные технологии
- Автоматическое компьютерное тестирование программ, разрабатываемых студентами

6.Учебно-методическое обеспечение самостоятельной работы студентов обучающихся по дисциплине «Программирование на языке высокого уровня».

Форма контроля и критерий оценок

В соответствии с учебным планом предусмотрен зачет в четвертом семестре.

Формы контроля: текущий контроль, промежуточный контроль по модулю, итоговый контроль по дисциплине предполагают следующее распределение баллов.

Текущий контроль

- Посещаемость занятий 5 баллов
- Выполнение 1 домашнейработы 10 баллов

Промежуточный контроль

По завершении модуля проводить письменный опрос 60 баллов

Темы для самостоятельного изучения.

№	Содержание дисциплины, самостоятельно изучаемой студентами	Формы контроля (контр.работа, лаб.занятия и т.д.)
1	**2**	**5**
1	Системное программное обеспечение. Классификация.	Практ. занятие

2	Кодирование экономической информации. Основы и методы защиты информации.	Практ. занятие
3	Периферийные устройства ПК. Оборудование для работы с мультимедиа и телекоммуникациями.	Практ. занятие
4	Системы счисления.	Практ. занятие
5	Понятие компьютерного вируса, виды вирусов. Признаки заражения компьютера вирусом. Наиболее популярные программы борьбы с компьютерными вирусами.	Практ. занятие
6	Эффективность работы программы; изменяемость программных средств; переносимость программных средств.	опрос.занятие.
7	Команды встроенного редактора системы TURBO Си++	опрос.занятие
8	вные конструкции алгоритмического языка TURBO Си++	Прак. занятие
9	Введение в систему типов языка TURBO Си++. Правила преобразования типов.	опрос
10	Базовые операции, запись выражений и стандартные функции в языке TURBO Си++	опрос
11	Программирование на языке TURBO Си++алгоритмов линейной структуры.	опрос
12	Программирование на языке TURBO Си++алгоритмов разветвляющейся структуры.	опрос
13	Программирование на языке TURBO Си++алгоритмов циклической структуры.	опрос
14	Действия с массивами.	Контр.работа

15	Программирование на языке TURBO Си++алгоритмов с использованием переменных строкового типа.	Контр.работа
16	Программирование на языке TURBO Си++ алгоритмов с использованием функций пользователя.	опрос
17	Указатели.	опрос
18	Структуры. Вложенные структуры. Дополнительные возможности.	опрос

Рекомендуемая литература.

а) основная литература:

1. Керниган, Б.В. Язык программирования С / Б.В. Керниган, Д.М. Ричи. - М. : Интернет-Университет Информационных Технологий, 2006. - 272 с. ; То же [Электронный ресурс]. - URL: http://biblioclub.ru/index.php?page=book&id=234039.

б) дополнительная литература:

1. Шилдт Г. Самоучитель С++, 3-е изд. СПб.: BHV – Санкт-Петербург, 2002.
2. Грегори К. Использование VisualC++ 6. Специальное издание. – М.: СПб.; К.: Издательский дом «Вильямс», 2000.
3. Тихомиров Ю.В. Самоучитель MFC. – СПб.: БХВ – Санкт-Петербург, 2000.

7. ФОНД ОЦЕНОЧНЫХ СРЕДСТВ ДЛЯ ПРОВЕДЕНИЯ ТЕКУЩЕГО КОНТРОЛЯ УСПЕВАЕМОСТИ, ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ ПО ИТОГАМ ОСВОЕНИЯ ДИСЦИПЛИНЫ.

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

В результате освоения дисциплины «Программирование на языках высокого уровня» ООП по специальности 09.03.02 Информационные системы и технологии обучающийся должен овладеть следующими результатами обучения по дисциплине:

Компетенция	Знания, умения, навыки	Процедура освоения
ПК-33	способностью составлять инструкции по эксплуатации информационных систем	**Знать:** Концепцию императивного программирования, основные синтаксические конструкции языка, методы программирования **Уметь:** писать и отлаживать программы на языке С++, осуществлять выбор и анализ соответствующих алгоритмов.

7.2. Описание показателей и критериев оценивания компетенций, описание шкал оценивания.

ПК-33
Схема оценки уровня формирования компетенции
«способность составлять инструкции по эксплуатации информационных систем».

Уровень	Показатели (что обучающийся должен продемонстрировать)	Оценочная шкала		
		Удовлетворительно	Хорошо	Отлично
Пороговый	способностью составлять инструкции по эксплуатации информационных систем	Знает, но допускает ошибки при определении концепций императивного программирования, основных синтаксических конструкций языка, методов программирования .	Достаточно хорошо знает и умеет писать и отлаживать программы на языке С++, осуществлять выбор и анализ соответствующих алгоритмов	Свободно владеет навыками описания алгоритмов, методами анализа функционирования разработанных программных продуктов

7.3. Типовые контрольные задания или иные материалы

ПРИМЕРЫ КОНТРОЛЬНО-ИЗМЕРИТЕЛЬНЫХ МАТЕРИАЛОВ

№Текст тестовых материалов

№Вопрос 1
Q: Разделение программы на функции:
1: является ключевым методом объектно-ориентированного программирования;
2: упрощает представление программы;
3: сокращает размер программного кода;
4: ускоряет процесс выполнения программы.
№Вопрос 2
Q: После имени функции ставятся …….
№Вопрос3
Q: Первая функция, вызываемая при запуске программы, это функция …
№Вопрос 4
Q: Конструкция C++, указывающая компьютеру выполнить действие, называется ……
№Вопрос 5
Q: Выражение:
1: всегда приводит к вычислению значения;
2: является способом высказывания программы;
3: всегда происходит вне функции;
4: является частью оператора.
№Вопрос 6
Q: Укажите размер в байтах переменных следующих типов в 32-битной системе: тип int …4..; тип longdouble …10.; тип float 4….; тип long ….4
№ВОПРОС 7
Q: Истинно ли следующее утверждение: переменная типа char может хранить значение 301?
1: да;
2: нет.
№Вопрос 8
Q: Истинно ли следующее утверждение: в операции присваивания величина, стоящая слева от знака равенства, всегда равна величине, стоящей справа от знака равенства?
1: истинно;
2: ложно.
№Вопрос 9
Q: Какой заголовочный файл нужно включить в исходный текст, чтобы использовать объекты cin и cout?
1: iomanip;
2: iostream;
3: fstream;
4: process;
5: ostream.

№Вопрос 10
Q: Напишите оператор, который получает с клавиатуры числовое значение и присваивает его переменной temp :
№Вопрос 11
Q:Какой заголовочный файл нужно включить в исходный текст, чтобы использовать манипулятор setw?
1: iomanip;
2: iostream;
3: fstream;
4: process;
5: ostream.
№Вопрос 12
Q: Верно или неверно следующее утверждение: нет никаких препятствий к использованию переменных разного типа в одном арифметическом выражении?
1: неверно
2: верно
№Вопрос 13
Q: Значение выражения 11 % 3 равно :2
№Вопрос 14
Q: Напишите оператор, увеличивающий значение переменной temp на 23 с одновременным присваиванием:
№Вопрос 15
Q: На какую величину увеличивает значение переменной операция инкремента?
V: 1 V:
№Вопрос 16
Q: Какие значения выведут на экран два указанных оператора cout<< var1--; cout<< ++var1;, если начальное значение переменной var1 равно 20? V:20 20 V:
№Вопрос 17
Q: Коды библиотечных функций содержатся в V: библиотечных V: файлах.
№Вопрос 18
Q: Каждая программа на языке C++ содержит функцию
1. head()
2.primary()
3. main()
4.prime()
5.major()
№Вопрос 19
Q:Оператор инкремента (++)
1. увеличивает значение переменной на единицу
2.увеличивает значение переменной на два
3.уменьшает значение переменной на единицу
4.уменьшает значение переменной на два
5не существует в языке C++
№Вопрос 20

Q: Нелогическим является оператор
1.&&
2. =
3. ||
4. !
№Вопрос 21
Q: Результат выполнения программы
#include<iostream>
int main()
{ int a = 5*3; float b=1.5f; b + = --a/2; cout<<b; return 1; }
1.8.50
2. 9.00
3.8.00
4.9.50
5.7.50
№Вопрос 22
Q: "<<"
1. оператор вывода в языке C++
2.встроенная функция
3.функция из стандартной библиотеки
4.макроопределение
5.нет правильного ответа
№Вопрос 23
Q: Лексема
1.заключительный оператор в программе
2. единица текста программы, которая при компиляции воспринимается как единое целое
3.первый оператор в программе
4.заголовок программы
5.фамилия создателя языка C++
№Вопрос 24
Q: Тип результата при сложении переменных типа short
1.short
2.int
3.unsigned
4.long
5.float
№Вопрос 25
Q: Фрагменткодаint a=3, b=4, c; c=a+++--b; cout<<"a="<<a<<", b="<<b<<", c="<<c; напечатает:
1.a=3, c=3, b=6
2.выражение некорректно
3.a=4, b=4, c=8
4.a=4, b=3, c=6

5.a=4, b=4, c=7
№Вопрос 26
Фрагменткодаint a=3,b=4, c=3; c=b+++--c; cout<<"a="<<a<<", b="<<b<<", c="<<c;
напечатает:
1.a=3, c=4, b=6
2.выражение некорректно
3. a=3, b=5, c=6
4.a=4, b=3, c=2
5.a=4, b=4, c=2
№Вопрос 27
Фрагменткодаint a=3,b=4, c=3; c=b+++--c+a++; cout<<"a="<<a<<", b="<<b<<",
c="<<c; напечатает:
1.a=3, c=2, b=4
2.выражение некорректно
3. a=4, b=5, c=9
4.a=3, b=3, c=2
5.a=4, b=6, c=2
№Вопрос 28
Q: Фрагментпрограммыinti = 4; int x = 6; double z; z = x / i; cout<<"z="<< z;
напечатает:
1.z=0.00
2.z=1.00
3.z=1.50
4.z=2.00
5.z=NULL
№Вопрос 29
Q: Фрагментпрограммыinti = 4; int x = 6; double z = 2; z+= x++/i++; cout<<"z=" z;
напечатает:
1.z=3.40
2. z=3.00
3.z=1.40
4.выражение некорректно
5.z=NULL
№Вопрос 30
Q: Выполнение программы на языке C++ начинается с
1.1-ой строки;
2.1-ой функции;
3. функции main;
4.нет правильного ответа
№Вопрос 31
Q: Файл с расширением obj содержит ….
1. исходный текст программы;
2.библиотечные функции;
3.исполняемую программу;

4. объектный код программы.
№Вопрос 32
Q: При правильном выполнении программы в операционную систему передается
1. нулевой результат;
2.ненулевой результат;
3.отрицательный результат;
4.нет правильного ответа
№Вопрос 33
Q: Не ключевое слово языка С++
1.break;
2.class;
3. go;
4.static;
5.this.
№Вопрос 34
Q: Операция = = относится к
1.операциям сдвига;
2. операциям сравнения;
3.операциям присваивания;
4.логическим операциям ;
5.нет правильного ответа
№Вопрос 35
Q: Отдельно стоящий символ "точка с запятой" считается
1.ошибкой;
2.разделителем;
3. пустым оператором;
4.концом программы;
5.концом строки.
№Вопрос 36
Q: Верное преобразование (если такое имеется), гарантирующее сохранение точности и неизменности численного значения:
1.signed char -> short -> char -> long ->int
2. float -> double -> long double
3.unsigned char -> unsigned short -> unsigned int -> double
4. Правильного преобразования нет.
№Вопрос 37
Q: Неверное утверждение:
1.программа на языке С++ - это совокупность функций, каждая из которых должна быть определена или по крайней мере описана до ее использования в конкретном модуле программы;
2.функция main обеспечивает создание точки входа в откомпилированную программу;

3.так же как и при определении массивов, не относящихся к динамическим, можно выполнить инициализацию динамического массива, поэтому при выделении памяти размеры такого массива можно явно не указывать;
4.всем именам функций программы по умолчанию присваивается класс памяти extern;
№Вопрос 38
Q: Ошибочное утверждение:
1.C++ обеспечивает "строгий контроль типов";
2.строгое согласование по типам между формальными и фактическими параметрами требует, чтобы в модуле до первого обращения к функции было помещено либо ее описание, либо ее определение.
3.описание функции - это ее прототип,
4.при обращении к функции, фактические параметры заменяются формальными.
№Вопрос 39
Q: Правильный вариант программы, выводящей на экран "HelloWorld":
1.#include <iostream> void main() { cout<<"Hello World"; return; }
2. #include <iostream>intmain() { cout<<"Hello World"; return 0; }
3.#include <iostream>; int main() { cout<<"Hello World"; return 0; }
4.include<iostream> void main() { cout<<"Hello World"; return 0; }
№Вопрос 40
Q: Правильный вариант программы, выводящей на экран "I love C++":
1. #include <iostream> void main() { cout<<"I love C++"; return; }
2. #include <iostream>int main() { cout<<"I love C++"; return 0; }
3.#include <iostream> ; int main() { cout<<"I love C++"; return 0; }
4.#include <iostream> ; void main() { cout<<"I love C++"; return 0; }
№Вопрос 41
Q: Преимущество использования ключевого слова const:
1.константу, определенную с помощью const, можно изменять во время работы;
2.к константе, определенной с помощью const, можно применить операции инкремента и декремента;
3.константа, определенная с помощью const, доступна в других модулях программы;
4. константа, определенная с помощью const, имеет тип, и компилятор может проследить за ее использованием в соответствии с объявленным типом.
№Вопрос 42
Q: Команда …… вставляет в программу заранее подготовленные тексты из включаемых файлов.
№Вопрос 43
Q: Какие основные области применения языка C++?
1.системное программирование
2.прикладное программирование
3. системное и прикладное программирование
№Вопрос 44
Q: Компилятор языка C++:

1.переводит текст программы в машинные инструкции
2.выполняет программу
3. форматирует текст программы так, чтобы его было удобно читать
№Вопрос 45
Q: Комментарий в программе на C++
1.содержит указания компилятору по настройке программы
2. содержит пояснения к тексту и не оказывает влияния на выполнение программы
3.должен содержать допустимые аргументы программы
№Вопрос 46
Q: При выходе из функции main()
1.программа повторяется с теми же аргументами
2. программа завершается
3. выполняется функция finish, определенная программистом
№Вопрос 47
Q: Укажите неправильный идентификатор:
1.AB_D1
2. 1xd
3.z1d8_14f3
№Вопрос 48
Q: Проверка типов переменной
1. осуществляется во время компиляции
2.осуществляется во время выполнения программы
3.производится, если при объявлении переменных был задан тип
№Вопрос 49
Q: Выберите правильное объявление константы pi:
1. const float pi = 3.14;
2.float pi = (const) 3.14;
3.const float pi; pi = 3.14;
№Вопрос 50
Q: Какое из приведенных имен является недопустимым в C++?
1. Abc_87F$_7
2.x03488erJJJ_
3. _a$$$error
4. xb___@
5.r13
6.OOP

7.4. Методические материалы, определяющие процедуру оценивания знаний, умений, навыков и (или) опыта деятельности, характеризующих этапы формирования компетенций.

а) **Критерии оценивания компетенций (результатов).**

Программой дисциплины в целях проверки прочности усвоения материала предусматривается проведение различных форм контроля:
1. «Входной» контроль определяет степень сформированностизнаний, умений и навыков обучающегося, необходимым для освоения дисциплины и приобретенным в результате освоения предшествующих дисциплин.
2. Тематический контрольопределяет степень усвоения обучающимися каждого раздела (темы в целом), их способности связать учебный материал с уже усвоенными знаниями, проследить развитие, усложнение явлений, понятий, основных идей.
3. Межсессионная аттестация– рейтинговый контроль знаний студентов, проводимый в середине семестра.
4. Рубежной формой контроля является зачет. Изучение дисциплины завершается зачетом, проводимым в виде письменного опроса с учетом текущего рейтинга .

Рейтинговая оценка знаний студентов проводится по следующим критериям:

Вид оцениваемой учебной работы студента	Баллы за единицу работы	Максимальное значение
Посещение всех лекции	макс. 5 баллов	5
Присутствие на всех практических занятиях	макс. 5 баллов	5
Оценивание работы на семинарских, практических, лабораторных занятиях	макс. 10 баллов	10
Самостоятельная работа	макс. 40 баллов	40
Итого		60

Неявка студента на промежуточный контроль в установленный срок без уважительной причины оценивается нулевым баллом. Повторная сдача в течение семестране разрешается.
Дополнительные дни отчетности для студентов, пропустивших контрольную работупо уважительной причине, подтвержденной документально, устанавливаютсяпреподавателем дополнительно.
Практические занятия, пропущенные без уважительной причины, должны бытьотработаны до следующей контрольной точки, если сдаются позже, то оцениваются в 1балл.
Студенты, набравшие от 51 до 100 баллов, получают зачет по дисциплине без проведения дополнительных испытаний, если сумма набранных баллов меньше 50, тостудент пишет итоговый тест по дисциплине в последнюю учебную неделю семестра.

Итоговой формой контроля знаний, умений и навыков по дисциплине является **(зачет)**. Зачет проводится в форме тестирования. При соответствии ответа учащегося на зачете более чем 60 % критериев из этого списка выставляется оценка «зачет», в случае несоответствия – «незачет».

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

а) основная литература:

1. Павловская, Татьяна Александровна C/C++. Программирование на языке высокого уровня : для магистров и бакалавров: учебник для вузов / Т. А. Павловская .- СПб. :Питер , 2011
2. Русакова, Н.А. Программрование [Электронный ресурс] : электронный учеб.-метод.комплекс / Н.А. Русакова; Кемеровский гос. ун-т, Кафедра ЮНЕСКО по новыминформационным технологиям. - Электрон.дан. - Кемерово :КемГУ, 2009. - 1 эл. опт.диск (CD-ROM) http://unesco.kemsu.ru/study_work/method.htm
3. Ашарина И.В. Объектно-ориентированное программирование в С++: лекции и упражнения. "Горячая линия-Телеком" Издательство: 2012 Год: 320 стр., 2-е изд., стереотип.Издание:ttp://e.lanbook.com/books/element.php?pl1_cid=25&pl1_id=5115
4. Подбельский В.В. Фомин С.С. Курс программирования на языке Си "ДМК Пресс"Издательство:2012Год:384стр.

б) дополнительная литература

1. Задачник по программированию для математиков: Учебное пособие. Под редА.В.Старченко. – Томск: Изд-во Томского университета, 2001.
2. Кубенский А.А. Структуры и алгоритмы обработки данных: объектно-ориентированный подход и реализация на С++. – СПб.:БХВ-Петербург, 2004
3. Т. Кормен. и др. Алгоритмы. Построение и анализ.алгоритмов- МЦНМО, Москва,2001.
4. В.Н. Касьянов Программирование на языке Паскаль. Томск: Изд-во Томского университета, 2003г.

9. Перечень ресурсов информационно-телекоммуникационной сети «Интернет», необходимых для освоения дисциплины.

1. www.intuit.ru – Национальный Открытый Университете «ИНТУИТ»;
   http://e.lanbook.com/books/?p_f_1_temp_id=18&p_f_1_65=917&p_f_1_63=&p_f_1_67= - электронно-библиотечная система, издательство «Лань»;
2. www.elibrary.ru – научная электронная библиотека;

3. http://www.edu.ru/modules.php?op=modload&name=Web_Links&file=index&l_op=viewlink&cid=2720 – федеральный портал российского профессионального образования: Математика и естественнонаучное образование.

10.Методические указания для обучающихся по освоению дисциплины.

К современному специалисту общество предъявляет достаточно широкий переченьтребований, среди которых немаловажное значение имеет наличие у выпускников определенных способностей и умения самостоятельно добывать знания из различных источников,систематизировать полученную информацию, давать оценку конкретной финансовой ситуации. Формирование такого умения происходит в течение всего периода обучения через участие студентов в практических занятиях, выполнение контрольных заданий и тестов, написание курсовых и выпускных квалификационных работ. При этом самостоятельная работа студентов играет решающую роль в ходе всего учебного процесса.

Советы по планированию и организации времени, необходимого для изучения дисциплины.

Рекомендуется следующим образом организовать время, необходимое для изучения дисциплины:

Изучение конспекта лекции в тот же день, после лекции – 10-15 минут.

Изучение конспекта лекции за день перед следующей лекцией – 10-15 минут.

Изучение теоретического материала по учебнику и конспекту – 1 час в неделю.

Подготовка к практическому занятию – 2 часа.

Всего в неделю – 3 часа 25 минут.

Описание последовательности действий студента («сценарий изучения дисциплины»).

При изучении дисциплины необходимо не только выполнять практические задания попредмету, но и регулярно изучать теоретический материал.

1. После прослушивания лекции и окончания учебных занятий, при подготовке к практическим занятиям, нужно сначала просмотреть и обдумать текст лекции, прослушанной сегодня (10-15 минут).

2. При подготовке к лекции следующего дня, нужно просмотреть текст предыдущей лекции, подумать о том, какая может быть тема следующей лекции (10-15 минут).

3. Для выполнения лабораторной работы необходимо: Изучить учебные материалы, представленные в презентациях, выполнить предложенные преподавателем задания.

При выполнении упражнения или задачи нужно сначала понять, что требуется в задаче,какой теоретический материал нужно использовать, выбрать алгоритм решения задачи. Далее необходимо написать программу, провести ее отладку. Для исправления синтаксическихошибок необходимо обратиться к теоретическому материалу в лекциях, учебниках. Придальнейшей отладке программы необходимо пользоваться либо встроенными средствами,либо вставлять в программу дополнительные операторы вывода для возможности отслеживания полученных значений и локализации возможной ошибки. Для проверки правильностиработы программы необходимо составить достаточное количество тестовых заданий.

Рекомендации по использованию материалов учебно-методического комплекса.

Рекомендуется использовать методические указания по курсу программирования, текст лекций преподавателя (если он имеется), презентации лекций. Рекомендуется использоватьэлектронные учебно-методические пособия по программированию, имеющиеся на факультетском сервере.

Рекомендации по работе с литературой. Теоретический материал курса становитсяболее понятным, когда дополнительно к прослушиванию лекции и изучению конспекта, изучаются и учебники по программированию. Необходимая литература имеется как в библиотеке, так и в кабинете математики. Также по данному курсу имеется достаточно много учебныхматериалов в электронном

виде. При работе с литературой полезно одновременно читатьучебники нескольких авторов, после прочтения необходимо выполнить несколько заданий и упражнений самостоятельно, чтобы оценить степень усвоения материала.

Советы по подготовке к зачету. Дополнительно к изучению конспектов лекции необходимо пользоваться любым рекомендованным учебником по программированию.

При подготовке к зачету нужно изучить необходимый теоретический материал, повторитьосновные алгоритмы, и самостоятельно решить по нескольку типовых задач из каждой темы.

Советы по подготовке к экзамену. Дополнительно к изучению конспектов лекциинеобходимо пользоваться любым рекомендованным учебником по программированию. Необходимо повторить методы решения различных задач, самостоятельно решить часть из них.Внимательно ознакомиться с примерами тестовых заданий.

Указания по организации работы с контрольно-измерительными материалами,по выполнению домашних заданий.

При выполнении домашних заданий необходимо сначала прочитать основные понятия и теоремы по теме задания. При выполнении задания нужно сначала понять, что требуется в задаче, какой теоретический материал нужно использовать, выбрать алгоритм решения задачи, попытаться запрограммировать. Если это не далорезультатов, и необходимо рассмотреть решение подобных задач, и после этого попробоватьрешить предложенную задачу самостоятельно.

11.Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине, включая перечень программного обеспечения и информационных справочных систем.

1. Компьютерные классы с набором лицензионного базового программного обеспечениядля проведения лабораторных занятий;
2. MicrosoftVisualStudio (или CodeBloc) для выполнения лабораторных заданий
3. Лекционная мультимедийная аудитория для чтения лекций с использованием мультимедийных материалов.

4. Тестоваяпрограмма Test2000 для компьютерного тестирования.

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

При освоении дисциплины для выполнения лабораторных работ необходимы классыперсональных компьютеров с приложениями программирования на языках C/C++. Для проведения лекционных занятий, необходима мультимедийная аудитория с набором лицензионного базового программного обеспечения.

Лекционные занятия

- Видеопроектор, ноутбук, презентатор
- Подключение к сети Интернет

Практические занятия

- Видеопроектор, ноутбук
- Подключение к сети Интернет