МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
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 Информационных технологий БКС

Образовательная программа

09.03.04 Программная инженерия

Профиль подготовки:
Разработка программно-информационных систем

Уровень высшего образования:
бакалавриат

Форма обучения
очная

Статус дисциплины:
Входит в часть, формируемую участниками образовательных отношений

Махачкала, 2021

Рабочая программа дисциплины «Программирование на языке высокого уровня» составлена в 2021году в соответствии с требованиями ФГОС ВО - *бакалавриат* по направлению подготовки 09.03.04 Программная инженерия от «19» сентября 2017г. №922.

Разработчик(и): кафедра ИТиБКС, Ахмедова З.Х, доцент, кандидат ф-м. наук

Рабочая программа дисциплины одобрена:
на заседании кафедры ИТиБКС от «25» 05 2021 г., протокол № 11
Зав. кафедрой _____ Ахмедова З.Х.

на заседании Методической комиссии факультета ИиСТ от «28» апреля 2021г., протокол № 11.

Председатель _____ Бакмаев А.Ш..

Рабочая программа дисциплины согласована с учебно-методическим управлением «____» 9 июня 2021г. _____
(подпись)

Аннотация рабочей программы дисциплины

Дисциплина «Программирование на языке высокого уровня» входит в *часть, формируемую участниками образовательных отношений обр*азовательной программы *бакалавриата* по направлению 09.03.04 Программная инженерия
Дисциплина реализуется на факультете ИиИТ кафедрой _ИТиБКС.

Содержание дисциплины охватывает круг вопросов, связанных программированием на языках высокого уровня. Очевидно, что применение объектно-ориентированного подхода делает программы понятнее, надежнее и проще в использовании.

Дисциплина нацелена на формирование следующих компетенций выпускника: профессиональных – ПК-10.
Преподавание дисциплины предусматривает проведение следующих видов учебных занятий: *лекции, лабораторные занятия, самостоятельная работа.*

Рабочая программа дисциплины предусматривает проведение следующих видов контроля успеваемости в форме – *устный опрос, контрольная работа, коллоквиум в форме тестирования, кейс-задачи* и промежуточный контроль в форме *экзамена*.

Объем дисциплины 11 зачетных единиц, в том числе в академических часах по видам учебных занятий.

Объем дисциплины в очной форме

Семестр	Учебные занятия в том числе:					СРС, в том числе экзамен	Форма промежуточной аттестации (зачет, дифференцированный зачет, экзамен
	всего	Контактная работа обучающихся с преподавателем					
		всего	из них				
			Лекции	Лабораторные занятия	Практические занятия		
2	144	50	18	32		94	экзамен
3	144	54	18	36		90	экзамен
4	108	56	28	28		52	экзамен

1. Цели освоения дисциплины

Целью освоения дисциплины «Программирование языке высокого уровня» - базируется на освоение общих принципов алгоритмизации и разработки программ для ЭВМ, формирование способности осваивать методики использования программных средств для решения практических задач; получение знаний и навыков программирования на языке высокого уровня, самостоятельное приобретение с помощью информационных технологий и использование в практической деятельности новых знаний и умений.

2.Место дисциплины в структуре ОПОП бакалавриата

Дисциплина «Программирование языке высокого уровня»входит в *часть, формируемую участниками образовательных отношений* образовательной программы *бакалавриата* по направлению 09.03.04. Программная инженерия.

Для успешного освоения дисциплины студенты должны владеть знаниями, умениями и компетенциями, полученными ранее и при параллельном изучении дисциплин математического и естественнонаучного циклов: «Физика», «Математика», которые формируют необходимые для изучения программирования способности к обобщению и анализу информации, знания математического анализа и алгоритмов, структурных блоков ЭВМ, способов представления данных в ЭВМ, способность использовать персональный компьютер и системы программирования для разработки программного обеспечения, готовность понимать актуальность совершенствования языков программирования.

Без освоения дисциплины «Программирование языке высокого уровня» невозможна дальнейшая успешная подготовка студентов по направлению 09.03.04 Программная инженерия. Дисциплина «Программирование языке высокого уровня» определяет саму возможность изучения практически всех последующих дисциплин, поскольку в процессе изучения используются ЭВМ и языки высокого уровня, как средства и инструменты для исследований и получений результатов, для решения специализированных задач.

Дисциплина «Программирование языке высокого уровня» является основой следующих дисциплин: «Операционные системы», «Типы и структуры данных», «Базы данных», «Объектно-ориентированное программирование», «Технология программирования», «Компьютерная графика» и ряде других дисциплин, связанных с изучением или использованием программного обеспечения ЭВМ.

3. Компетенции обучающегося, формируемые в результате освоения дисциплины (перечень планируемых результатов обучения).

Код и наименование компетенции из ФГОС ВО	Код и наименование индикатора достижения компетенций	Планируемые результаты обучения	Процедура освоения
ПК-10 . Владение навыками использования различных технологий разработки программного обеспечения	ИПК 10.1	. Знает современные технологии разработки программного обеспечения (структурное, Объектноориентированное) Знает современные технологии разработки программного обеспечения (структурное, Объектноориентированное)	**Круглый стол**
	ИПК 10.2.	Умеет использовать современные технологии разработки программного	**Письменный опрос**

		обеспечения Умеет использовать современные технологии разработки программного обеспечения	
	ИПК 10.3.	Имеет навыки использования современных технологий разработки программного обеспечения Имеет навыки использования современных технологий разработки программного обеспечения	**Деловая игра**

4. Объем, структура и содержание дисциплины.

4.1. Объем дисциплины составляет 11 зачетных единиц, 396 академических часов.

4.2. Структура дисциплины.

4.2.1. Структура дисциплины в очной форме

№ п/п	Разделы и темы дисциплины	Семестр	Неделя семестра	Виды учебной работы, включая самостоятельную работу студентов и трудоемкость (в часах)				Формы текущего контроля успеваемости *(по неделям семестра)* Форма промежуточной аттестации *(по семестрам)*
				Лекции	Практические занятия	Лабораторные занятия	Самостоятельная работа	
	I семестр							
	Модуль 1. Язык Python и его особенности							
1	Введение. Классификация языков программирования. Этапы создания ПО	1		2		2	14	устный опрос
2	Типы и структуры данных. Ввод и вывод в Python. Структура программы .	1				2	4	устный опрос
3	Встроенные типы объектов: Числа Строки Кортежи Списки Словари Множества			2		2	6	устный опрос
	Итого по модулю 1:			4		8	24	
	Модуль 2. Основные алгоритмы и их реализация на Python							
1	Линейные алгоритмы. Операции с числами и строками .			4		8	6	Коллоквиум
2	Ветвления и			2		8	16	Тест

	оператор выбора .							
	Итого по модулю 2:			6		8	22	
	Модуль 3. Графика в Python и задачи моделирования							
1	Создание приложений с GUI. Обзор графических библиотек: Tkinter, PyQ			4		4	10	Кейс-задача Контрольная работа
2	Пользовательские подпрограммы и моделирование. Модуль Tkinter			4		4	10	Кейс-задача Контрольная работа
	Итого по модулю 4:			8		8	20	
	Модуль 4							
	Модуль 6 . Подготовка к экзамену	1					36	экзамен
	ИТОГО за 1 семестр:			18		32	94	
	II семестр							
	Модуль 1. Языки и системы программирования.							
1	Введение в язык C++. Простые типы. Литералы. Операции. Типы данных и операции C++	2		2		2	12	устный опрос
2	Базовые конструкции структурного программирования C++, организация ввода/вывода.	2		2		2	14	устный опрос
	Итого за 1 модуль:			4		4	28	
	Модуль 2. Разветвленные алгоритмы							
1	Условные конструкции в C++. Условная операция.			2		2	12	устный опрос
2	Оператор множественного выбора			4		4	14	устный опрос
	Итого за 2 модуль:			6		6	24	
	Модуль 3. Организация циклов в C++							
1	Составные типы. Циклы.			4		2	4	устный опрос
2	Цикл с параметром			2		2	4	
3	Использование операторов цикла для решения			2		4	8	Контрольная работа

	прикладных задач							
	Итого за модуль 3:			8		8	16	
	Модуль 4							
	Подготовка к экзамену						36	
	ИТОГО за 2 семестр			18		36	90	
	III семестр							
	Модуль 1. Типы данных, создаваемые пользователем							
1	Структуры. Массивы структур			4		4	2	устный опрос
2	Объединения			4		4	8	устный опрос
3	Перечисления			4		4	2	устный опрос
	Итого за модуль 1			12		12	12	
	Модуль 2. Организация ввода и вывода. Файловая система.							
1	Стандартные файлы ввода и вывода. Текстовые файлы.			4		4		Коллоквиум
2	Основные методы обработки текстовых файлов.			4		4		Коллоквиум
3	Двоичные файлы			4		4		Коллоквиум
4	Аргументы командной строки			4			4	
	Итого за модуль 2			16		16	4	
	Модуль 3 Подготовка к экзамену.							
	Подготовка к экзамену						36	
	ИТОГО за III семестр			28		28	52	

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

4.3.1. Содержание лекционных занятий по дисциплине.

1 семестр.
Модуль 1-2.
Тема 1.
1.Введение в Python
Тема 2.
2. Установка и настройка Python. Среды разработки, работа с jupyter notebooks. Как читать документацию. Принципы работы веба.
Модуль 3
Тема 1.
1. Типы данных и переменные
Работа с данными различных типов: целые числа, дробные числа, строки и логические значения.
Тема 2.
2. Переменные и константы. Арифметические операции в Python.

Тема 3.
3.Функции len(), sum(), count(), type(), print(), input().
Модуль 4.
Тема 1.
1. Структуры данных и условные выражения.
Структуры данных: списки, кортежи, множества.
Тема 2.
2.Условные выражения.
Конструктор if – elif – else. Операторы or, not, in, and.
Модуль 5.
Тема 1
1. Словари и функции
Типы данных: словари. Методы работы со словарями. Определение и вызов функции. Обязательные и необязательные аргументы. Правила написания функций.
Тема 2
2. Циклы
Понятие итерации. Циклы for и while. Операции break и continue. Итерация по ключам и значениям словаря. Функция range().
Тема 3.
3. Работа с файлами
Типы файлов. Режимы доступа к файлу. Чтение файла и запись в файл. Пути к файлам и папкам. Библиотека os.
Тема 4.
4. Строки и генераторы списков
Продвинутые инструменты работы со строками. Функции split(), join(), strip(), replace(). Функции для работы с регистром и типами символов в строке. Генераторы списков.
II семестр
1 –2 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;

3-4 модуль

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;

5 модуль

Тема 9. Основы создания элементов управления ActiveX.

III семестр.

1-2 модуль

Тема 1.

1. Структуры. Массивы структур
2. Объединения
3. Перечисления
4. Стандартные файлы ввода и вывода. Текстовые файлы.
5. Основные методы обработки текстовых файлов.
6. Двоичные файлы
7. Аргументы командной строки

3-4 модуль

Тема 2.

1. Объявление указателей. Разыменование указателей.
2. Ссылки. Резервирование памяти в куче
3. Указатели и массивы
4. Строковые указатели
5. Указатели и структуры

5 модуль

Тема 3.

1. Формирование очереди
2. Формирование стека
3. Двусвязные списки

4.3.2. Содержание лабораторно-практических занятий по дисциплине.
I-II семестр

1. Набрать программу, которая выводит на экран ваши имя и фамилию.

2. Написать программу, которая вычисляет площадь поверхности и объем цилиндра (по формулам $S = 2nr(r + h)$, $V = nr^2h$. Для форматного ввода-вывода используйте функции scanfQ и printfQ, в ответе сохраните 5 цифр после десятичной точки.

3. Написать программу, которая вычисляет объем параллелепипеда.

Далее приводится рекомендуемый вид экрана

Вычисление объема параллелепипеда

Введите исходные данные:

Длина(см) -> 9.00

Ширина(см) -> 7.50

Высота(см) -> 5.00

Объем: 337.50 куб.см.

1.5. Напишите программу, которая вычисляет корни квадратного уравнения $ax^2 + bx + c = 0$ (в предположении, что $a = 0$) в зависимости от знака дискриминанта $D = b^2 — 4ac$: при $D < 0$ выдается сообщение, что корней нет; при $D = D > 0$ вычисляются соответственно один или два корня и выдаются на экран.

1.6. Напишите программу, вычисляющую сумму $S(x)=\sum_{k=0}^{\infty}\frac{x^k}{k!}$ с заданной точностью eps. Результат для некоторого конкретного значения x сравнить со значением e^x. Для вычисления e^x используем стандартную функцию *exp* из библиотеки math.h.

1.7*. Разработайте алгоритм и составьте программу, которая бы всякое натуральное число N, не превосходящее 2000, представляла бы римскими цифрами (в соответствии с обозначениями (1) — см. подраздел 2.1).

1.8*. Напишите программу, которая складывала бы двоичные, восьмеричные и шестнадцатеричные числа.

1.9*. Напишите программу перевода натурального десятичного числа в двоичную, восьмеричную и шестнадцатеричную форму .

1.10*. Напишите программу, которая бы переводила число из двоичной, восьмеричной и шестнадцатеричной формы в десятичную.

1.11.Напишите программу, реализующую формулу прямоугольников приближенного вычисления определенного интеграла

**** помечены задания повышенной сложности***

Темы 1-5.

1. Задача с тремя массивами
2. Обработка матриц
3. Работа с двоичными файлами
4. Работа со строками
5. Указатели на функции

1. Задача с тремя массивами

В каждом варианте необходимо обработать три массива **разного размера**. Необходимо написать функцию, которая выполняет требуемые действия, применить её ко всем трём массивам и сравнить полученные результаты.

Ввод осуществляется из файлов с использованием аргументов функции *main*. Вывод – на экран или в файл с обязательным выводом исходных данных.

Пример программы см. в *лекции 3*.

Задача с тремя массивами

В каждом варианте необходимо обработать три массива **разного размера**. Необходимо написать функцию, которая выполняет требуемые действия, применить её ко всем трём массивам и сравнить полученные результаты.

Ввод осуществляется из файлов с использованием аргументов функции *main*. Вывод – на экран или в файл с обязательным выводом исходных данных.

Пример программы см. в *лекции 3*.

1. Определить в каком массиве больше среднее арифметическое элементов, меньших заданного числа. Если в двух или трёх массивах значения среднего арифметического совпадают, вывести соответствующее сообщение.
2. Определить в каком массиве меньше среднее арифметическое элементов, больших заданного числа. Если в двух или трёх массивах значения среднего арифметического совпадают, вывести соответствующее сообщение.
3. Определить в каком массиве больше количество элементов, меньших заданного числа. Если в двух или трёх массивах количества искомых элементов совпадают, вывести соответствующее сообщение.
4. Определить в каком массиве меньше количество элементов, больших заданного числа. Если в двух или трёх массивах количества искомых элементов совпадают, вывести соответствующее сообщение.
5. Определить в каком массиве больше минимум элементов, больших заданного числа. Если в двух или трёх массивах минимумы совпадают, вывести соответствующее сообщение.
6. Определить в каком массиве меньше максимум элементов, меньших заданного числа. Если в двух или трёх массивах максимумы совпадают, вывести соответствующее сообщение.
7. Определить в каком массиве больше сумма элементов, попадающих в заданный диапазон. Если в двух или трёх массивах суммы совпадают, вывести соответствующее сообщение.
8. Определить в каком массиве меньше сумма элементов, не попадающих в заданный диапазон. Если в двух или трёх массивах суммы совпадают, вывести соответствующее сообщение.
9. Определить в каком массиве больше произведение элементов, не попадающих в заданный диапазон. Если в двух или трёх массивах произведения совпадают, вывести соответствующее сообщение.
10. Определить в каком массиве меньше произведение элементов, попадающих в заданный диапазон. Если в двух или трёх массивах произведения совпадают, вывести соответствующее сообщение.
11. Определить в каком массиве меньше среднее арифметическое элементов, меньших заданного числа. Если в двух или трёх массивах значения среднего арифметического совпадают, вывести соответствующее сообщение.
12. Определить в каком массиве больше среднее арифметическое элементов, больших заданного числа. Если в двух или трёх массивах значения среднего арифметического совпадают, вывести соответствующее сообщение.
13. Определить в каком массиве меньше количество элементов, меньших заданного числа. Если в двух или трёх массивах количества искомых элементов совпадают, вывести соответствующее сообщение.
14. Определить в каком массиве больше количество элементов, больших заданного числа. Если в двух или трёх массивах количества искомых элементов совпадают, вывести соответствующее сообщение.
15. Определить в каком массиве меньше минимум элементов, больших заданного числа. Если в двух или трёх массивах минимумы совпадают, вывести соответствующее сообщение.

16. Определить в каком массиве больше максимум элементов, меньших заданного числа. Если в двух или трёх массивах максимумы совпадают, вывести соответствующее сообщение.
17. Определить в каком массиве меньше сумма элементов, попадающих в заданный диапазон. Если в двух или трёх массивах суммы совпадают, вывести соответствующее сообщение.
18. Определить в каком массиве больше сумма элементов, не попадающих в заданный диапазон. Если в двух или трёх массивах суммы совпадают, вывести соответствующее сообщение.
19. Определить в каком массиве меньше произведение элементов, не попадающих в заданный диапазон. Если в двух или трёх массивах произведения совпадают, вывести соответствующее сообщение.
20. Определить в каком массиве больше произведение элементов, попадающих в заданный диапазон. Если в двух или трёх массивах произведения совпадают, вывести соответствующее сообщение.
21. Определить в каком массиве больше среднее арифметическое элементов, меньших заданного числа. Если в двух или трёх массивах значения среднего арифметического совпадают, вывести соответствующее сообщение.
22. Определить в каком массиве меньше среднее арифметическое элементов, больших заданного числа. Если в двух или трёх массивах значения среднего арифметического совпадают, вывести соответствующее сообщение.
23. Определить в каком массиве больше количество элементов, меньших заданного числа. Если в двух или трёх массивах количества искомых элементов совпадают, вывести соответствующее сообщение.
24. Определить в каком массиве меньше количество элементов, больших заданного числа. Если в двух или трёх массивах количества искомых элементов совпадают, вывести соответствующее сообщение.
25. Определить в каком массиве больше минимум элементов, больших заданного числа. Если в двух или трёх массивах минимумы совпадают, вывести соответствующее сообщение.

2. Обработка матриц

При выполнении этого задания необходимо написать **две** функции (кроме функции ввода матрицы). Одна из этих функций должна обрабатывать **матрицу целиком**. Другая функция должна обрабатывать **одномерный массив**. В качестве этого одномерного массива передаётся **одна строка матрицы.**

Ввод осуществляется из файлов с использованием аргументов функции *main*. Вывод – на экран или в файл с обязательным выводом исходных данных.

Пример программы см. в *лекции 3*.

Обработка матриц

При выполнении этого задания необходимо написать **две** функции (кроме функции ввода матрицы). Одна из этих функций должна обрабатывать **матрицу целиком**. Другая функция должна обрабатывать **одномерный массив**. В качестве этого одномерного массива передаётся **одна строка матрицы.**

Ввод осуществляется из файлов с использованием аргументов функции *main*. Вывод – на экран или в файл с обязательным выводом исходных данных.

Пример программы см. в *лекции 3*.

1. Даны две матрицы разного размера. Для той из матриц, в которой больше максимальный элемент, найти максимальный элемент в каждой строке.
2. Даны две матрицы разного размера. Для той из матриц, в которой больше максимальный элемент, проверить наличие положительных элементов в каждой строке.
3. Даны две матрицы разного размера. Для той из матриц, в которой больше максимальный элемент, найти количество положительных элементов в каждой строке.
4. Даны две матрицы разного размера. Для той из матриц, в которой больше максимальный элемент, найти сумму положительных элементов в каждой строке.
5. Даны две матрицы разного размера. Для той из матриц, в которой больше максимальный элемент, найти среднее арифметическое ненулевых элементов в каждой строке.
6. Даны две матрицы разного размера. Для той из матриц, в которой есть элементы, равные 0, найти минимальный элемент в каждой строке.
7. Даны две матрицы разного размера. Для той из матриц, в которой есть элементы, равные 0, проверить наличие отрицательных элементов в каждой строке.
8. Даны две матрицы разного размера. Для той из матриц, в которой есть элементы, равные 0, найти количество отрицательных элементов в каждой строке.
9. Даны две матрицы разного размера. Для той из матриц, в которой есть элементы, равные 0, найти произведение ненулевых элементов в каждой строке.
10. Даны две матрицы разного размера. Для той из матриц, в которой есть элементы, равные 0, найти среднее арифметическое положительных элементов в каждой строке.
11. Даны две матрицы разного размера. Для той из матриц, в которой меньше количество нулевых элементов, найти минимальный элемент в каждой строке.
12. Даны две матрицы разного размера. Для той из матриц, в которой меньше количество нулевых элементов, проверить наличие отрицательных элементов в каждой строке.
13. Даны две матрицы разного размера. Для той из матриц, в которой меньше количество нулевых элементов, найти количество отрицательных элементов в каждой строке.
14. Даны две матрицы разного размера. Для той из матриц, в которой меньше количество нулевых элементов, найти произведение ненулевых элементов в каждой строке.
15. Даны две матрицы разного размера. Для той из матриц, в которой меньше количество нулевых элементов, найти среднее арифметическое положительных элементов в каждой строке.
16. Даны две матрицы разного размера. Для той из матриц, в которой меньше среднее арифметическое положительных элементов, найти минимальный элемент в каждой строке.
17. Даны две матрицы разного размера. Для той из матриц, в которой меньше среднее арифметическое положительных элементов, проверить наличие отрицательных элементов в каждой строке.
18. Даны две матрицы разного размера. Для той из матриц, в которой меньше среднее арифметическое положительных элементов, найти количество отрицательных элементов в каждой строке.
19. Даны две матрицы разного размера. Для той из матриц, в которой меньше среднее арифметическое положительных элементов, найти произведение ненулевых элементов в каждой строке.
20. Даны две матрицы разного размера. Для той из матриц, в которой меньше среднее арифметическое положительных элементов, найти среднее арифметическое положительных элементов в каждой строке.
21. Даны две матрицы разного размера. Для той из матриц, в которой больше произведение ненулевых элементов, найти максимальный элемент в каждой строке.
22. Даны две матрицы разного размера. Для той из матриц, в которой больше произведение ненулевых элементов, проверить наличие положительных элементов в каждой строке.
23. Даны две матрицы разного размера. Для той из матриц, в которой больше произведение ненулевых элементов, найти количество положительных элементов в каждой строке.

24. Даны две матрицы разного размера. Для той из матриц, в которой больше произведение ненулевых элементов, найти сумму положительных элементов в каждой строке.
25. Даны две матрицы разного размера. Для той из матриц, в которой больше произведение ненулевых элементов, найти среднее арифметическое ненулевых элементов в каждой строке.

3.Работа с двоичными файлами

В двоичном файле поменять местами две записи с заданными номерами. Обязательно проверить, что записи с такими номерами существуют в файле. Следует читать не весь файл, а только нужные записи – в двоичном файле возможен прямой доступ (**никаких циклов!**). При выполнении этого задания необходимо преобразовать текстовый файл в двоичный, а также вывести на экран содержимое двоичного файла до обработки и после обработки. Не забывайте закрывать файлы.

Примеры см. в *лекции 4*.

1. Каждая запись представляет собой 2 символа.
2. Каждая запись представляет собой 2 числа типа *short int*.
3. Каждая запись представляет собой 2 числа типа *long int*.
4. Каждая запись представляет собой 2 числа типа *float*.
5. Каждая запись представляет собой 2 числа типа *double*.
6. Каждая запись представляет собой 3 символа.
7. Каждая запись представляет собой 3 числа типа *short int*.
8. Каждая запись представляет собой 3 числа типа *long int*.
9. Каждая запись представляет собой 3 числа типа *float*.
10. Каждая запись представляет собой 3 числа типа *double*.
11. Каждая запись представляет собой 4 символа.
12. Каждая запись представляет собой 4 числа типа *short int*.
13. Каждая запись представляет собой 4 числа типа *long int*.
14. Каждая запись представляет собой 4 числа типа *float*.
15. Каждая запись представляет собой 4 числа типа *double*.
16. Каждая запись представляет собой 5 символов.
17. Каждая запись представляет собой 5 чисел типа *short int*.
18. Каждая запись представляет собой 5 чисел типа *long int*.
19. Каждая запись представляет собой 5 чисел типа *float*.
20. Каждая запись представляет собой 5 чисел типа *double*.
21. Каждая запись представляет собой 6 символов.
22. Каждая запись представляет собой 6 чисел типа *short int*.
23. Каждая запись представляет собой 6 чисел типа *long int*.
24. Каждая запись представляет собой 6 чисел типа *float*.
25. Каждая запись представляет собой 6 чисел типа *double*.

6. **Работа со строками**

Для решения задачи вам нужно ввести строки с помощью функций *gets* или *fgets*, разбить строку на слова и выбрать нужные (функция *scanf* с форматом *%s* вводит строку до пробела, но в данной задаче вам **не нужно** так делать!). **Стандартные функции работы со строками НЕ использовать!** Обратите внимание, что во всех случаях слова могут разделяться **любым (!)** количеством символов, не относящихся к слову (будем считать, что к слову относятся большие и маленькие латинские буквы и цифры). Желательно всю обработку выполнить за один проход строки, хотя это возможно не во всех вариантах. Обязательно проверить работу программы на пустой строке и на строке, состоящей только из символов, не относящихся к слову.

Примеры см. в *лекции 5*.

1. Сформировать строку из тех же слов исходной строки в обратном порядке.
2. Сформировать строку заменой в исходной строке заданной подстроки на другую заданную подстроку (возможно разной длины).
3. Сформировать строку, добавляя к каждой заданной подстроке другую заданную подстроку.
4. Реверсировать буквы в каждом слове.
5. Сформировать строку из слов исходной строки в алфавитном порядке.
6. Сформировать строку из слов исходной строки в порядке увеличения количества символов в слове.
7. Сформировать строку из слов исходной строки в порядке уменьшения количества символов в слове.
8. Сформировать строку из слов исходной строки, содержащих повторяющиеся буквы.
9. Сформировать строку из слов исходной строки, в которых нет повторяющихся букв.
10. Сформировать строку, удалив из каждого слова исходной строки повторяющиеся в нем буквы.
11. Сформировать строку, «склеив» первое слово с последним, второе с предпоследним и т.д.
12. Сформировать строку, «склеив» первое слово со слово с номером (**n** + 1) / 2 + 1, второе – со словом с номером (**n** + 1) / 2 + 2 и т.д. (**n** – количество слов в строке).
13. Сформировать строку из тех же слов исходной строки в обратном порядке.
14. Сформировать строку заменой в исходной строке заданной подстроки на другую заданную подстроку (возможно разной длины).
15. Сформировать строку, добавляя к каждой заданной подстроке другую заданную подстроку.
16. Реверсировать буквы в каждом слове.
17. Сформировать строку из слов исходной строки в алфавитном порядке.
18. Сформировать строку из слов исходной строки в порядке увеличения количества символов в слове.
19. Сформировать строку из слов исходной строки в порядке уменьшения количества символов в слове.
20. Сформировать строку из слов исходной строки, содержащих повторяющиеся буквы.
21. Сформировать строку из слов исходной строки, в которых нет повторяющихся букв.
22. Сформировать строку, удалив из каждого слова исходной строки повторяющиеся в нем буквы.
23. Сформировать строку, «склеив» первое слово с последним, второе с предпоследним и т.д.
24. Сформировать строку, «склеив» первое слово со слово с номером (**n** + 1) / 2 + 1, второе – со словом с номером (**n** + 1) / 2 + 2 и т.д. (**n** – количество слов в строке).
25. Сформировать строку из тех же слов исходной строки в обратном порядке.
26. Сформировать строку заменой в исходной строке заданной подстроки на другую заданную подстроку (возможно разной длины).
27. Сформировать строку, добавляя к каждой заданной подстроке другую заданную подстроку.
28. Реверсировать буквы в каждом слове.
29. Сформировать строку из слов исходной строки в алфавитном порядке.
30. Сформировать строку из слов исходной строки в порядке увеличения количества символов в слове.

5. Указатели на функции

Переделать *задачу с тремя массивами* так, чтобы вспомогательная функция (не *main*) работала не с элементом массива X_i, а с выражением $f(X_i)$, где f – некоторая функция (одного аргумента!), указатель на которую передаётся через параметры. В качестве фактического параметра передавать указатель на стандартную библиотечную функцию и указатель на пользовательскую функцию. *Ввод/вывод производить из файла/в файл* (пример см. в *лекции 4*).

Про указатели на функции см. в *лекции 5*.

Ш семестр

Тема: «Функции языка С++».

1. Имеется следующий образец структуры:

```
struct box
{
char maker[40];
float height;
float width;
float length;
float volume;
};
```

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:

```
struct box
{
char maker[40];
float height;
float width;
float length;
float volume;
};
```

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 Ox.

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

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, на которой 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.

5. Образовательные технологии

В учебном процессе помимо традиционных форм проведения занятий используются лекции – визуализации, лекции – диалоги. Лабораторные занятия проводятся в компьютерном классе с использованием Интернет 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.

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 домашней работы 10 баллов
- Активность в системе Moodle 10 баллов

Промежуточный контроль
По завершении модуля проводить письменный опрос 60 баллов
Примерное распределение времени самостоятельной работы студентов

Вид самостоятельной работы	Примерная трудоёмкость, а.ч.		
	Очная	Очно-заочная	заочная
Текущая СРС			
работа с лекционным материалом, с учебной литературой	40	--	-
опережающая самостоятельная работа (изучение нового материала до его изложения на занятиях)	***40***	-	-
самостоятельное изучение разделов дисциплины	***20***	-	-
выполнение домашних заданий, домашних контрольных работ	***40***	-	-
подготовка к лабораторным работам, к практическим и семинарским занятиям	***20***	-	-
подготовка к контрольным работам, коллоквиумам, зачётам	***20***	-	-
подготовка к экзамену	***36***	-	-
Творческая проблемно-ориентированная СРС			
поиск, изучение и презентация информации по заданной проблеме, анализ научных публикаций по заданной теме		-	-
исследовательская работа, участие в конференциях, семинарах, олимпиадах		-	-
анализ данных по заданной теме, написание программ, составление моделей на основе исходных данных		-	-
Итого СРС:	***236***	-	-

7. Фонд оценочных средств для проведения текущего контроля успеваемости, промежуточной аттестации по итогам освоения дисциплины.

7.1. Типовые контрольные задания

Комплект заданий для промежуточного контроля.

№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 long double …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.макрОПОПределение
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: Фрагмент кода int 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
Фрагмент кода int 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
Фрагмент кода int 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: Фрагмент программы int i = 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: Фрагмент программы int i = 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 С++ 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.breaк;
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.

Кейс-задача

Задание(я):

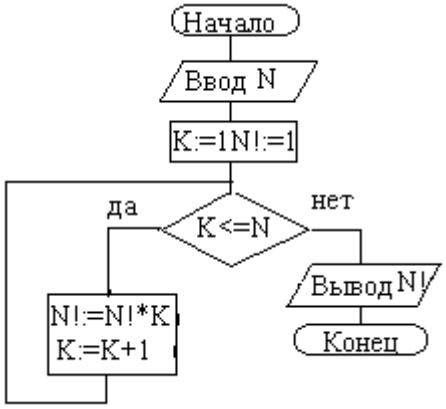
№1 .Кейс
№вопрос1
Дана программа. Что произойдёт в результате компиляции и выполнения данного кода?

```
#include <iostream>
using namespace std;
const int &fun()
{
static int a = 0;
return a;
}
```

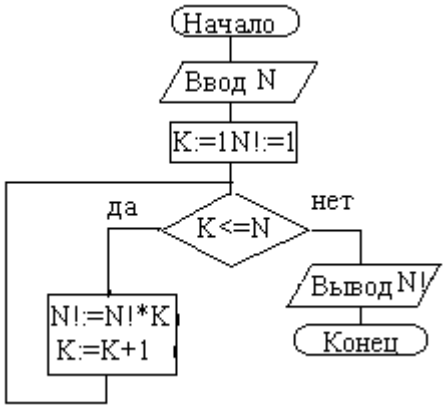
```
int main()
{
++fun();
cout << fun() << endl;
return 0;
}
```
№вопрос1
Дана программа. Произойдет ли ошибка компиляции, и если да, то почему?
```
#include <iostream>
using namespace std;
const int &fun()
{
static int a = 0;
return a;
}
int main()
{
++fun();
cout << fun() << endl;
return 0;
}
```
№вопрос1
Дана программа. Каково действие операции инкремента?
```
#include <iostream>
using namespace std;
const int &fun()
{
static int a = 0;
return a;
}
int main()
{
++fun();
cout << fun() << endl;
return 0;
}
```
№ 2 Кейс
№вопрос1
Дана программа. Что произойдёт в результате компиляции и выполнения следующего кода?
```
#include <iostream>
using namespace std;
int main(int argc, char *argv[])
{ cout << 1 / 2 ? 0 : 1 << endl;
return 0;
}
```
№вопрос1
Дана программа. У какого оператора меньший приоритет?
```
#include <iostream>
using namespace std;
int main(int argc, char *argv[])
{ cout << 1 / 2 ? 0 : 1 << endl;
return 0;
```

}
№вопрос1
Дана программа.
Данный код эквивалентен следующему (cout << 1 / 2) ? (0) : (1 << endl) или нет?
#include <iostream>
using namespace std;
int main(int argc, char *argv[])
{
cout << 1 / 2 ? 0 : 1 << endl;
return 0;
}
№3 Кейс
№вопрос1
Дана программа. Что будет выведено в результате компиляции и выполнения данного кода?
#include <iostream>
#include <stdio.h>
using namespace std;
int main(int argc, char *argv[])
{
cout << printf("boom!");
return 0;
}
№вопрос1
Дана программа. Что будет выведено в результате компиляции и выполнения данного кода?
#include <iostream>
#include <stdio.h>
using namespace std;
int main(int argc, char *argv[])
{
cout << printf("boom!");
return 0;
}
"boom!5
boom!
boom!true
boom!false
№вопрос2
Дана программа. Какое из утверждений верно?
#include <iostream>
#include <stdio.h>
using namespace std;
int main(int argc, char *argv[])
{
cout << printf("boom!");
return 0;
}
printf() возвращает int, равное количеству символов, выведенных в stdout.
cout по умолчанию синхронизирован с stdout
cout по умолчанию не синхронизирован с stdout
printf() возвращает int, не равное количеству символов, выведенных в stdout
№ 4 Кейс
№вопрос1

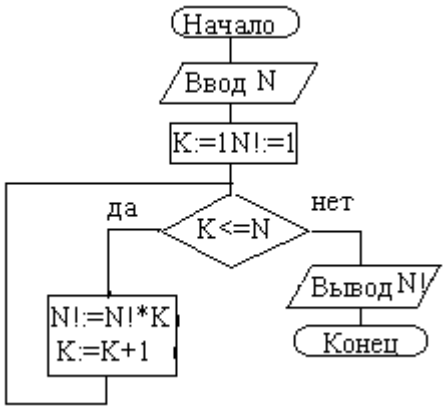
Задан алгоритм вычисления N факториала. На рисунке представлена блок-схема. При заданных исходных данных (N=3) определите результат выполнения алгоритма вычисления факториала



№вопрос1
Задан алгоритм вычисления N факториала. На рисунке представлена блок-схема. Алгоритмическая структура какого типа изображена на блок-схеме?



№вопрос1
Задан алгоритм вычисления N факториала. На рисунке представлена блок-схема.
Какой вид цикла представлен на блок-схеме?



Вопросы для коллоквиумов, собеседования
Раздел I. Основы языков C / C ++

1. Основы языков C / C ++: синтаксис языка, директивы препроцессора, консольное приложение.
2. Обзор C ++: объектно-ориентированное программирование - инкапсуляция, полиморфизм, наследование.
3. Классификация типов данных в C++. Арифметические типы данных
4. Классификация типов данных в C++. Арифметические типы данных. Преобразование типов в выражениях.

5. Переменные, константы, операторы и выражения.
6. Переменные, константы, операторы и выражения. Приоритеты операций и операторов в С ++.
7. Потоки и файлы; ввод, вывод, управляющие последовательности. Библиотечные функции

Раздел II. Составной тип

8. Составной тип – строки. Методы ввода строк.
9. Составные типы – структуры, объединения, перечисления и определяемые пользователем типы
10. Указатели и строки
11. Массивы. Одномерные и многомерные массивы
12. Функции пользователя в С++. Передача параметров в функции по значению и по ссылке

Вопросы к экзамену

1. Основы языков С / С ++: синтаксис языка, директивы препроцессора, консольное приложение.
2. Обзор С ++: объектно-ориентированное программирование - инкапсуляция, полиморфизм, наследование.
3. Классификация типов данных в С++. Арифметические типы данных
4. Классификация типов данных в С++. Арифметические типы данных. Преобразование типов в выражениях.
5. Переменные, константы, операторы и выражения.
6. Переменные, константы, операторы и выражения. Приоритеты операций и операторов в С ++.
7. Потоки и файлы; ввод, вывод, управляющие последовательности. Библиотечные функции
8. Функции: прототип, определение, реализация. Функция main ().
9. Функции: прототип, определение, реализация. Структура С++ -программы.
10. Логические операции. Операторы управления С++-программой
11. Составной тип – строки. Методы ввода строк.
12. Составные типы – структуры, объединения, перечисления и определяемые пользователем типы
13. Виды циклов в С++. Цикл с параметром.
14. Виды циклов в С++. Цикл с предусловием
15. Виды циклов в С++. Цикл с постусловием
16. Указатели и свободная память. Работа с памятью с помощью new и delete
17. Указатели и строки
18. Массивы. Одномерные массивы
19. Массивы. Одномерные и многомерные массивы
20. Функции пользователя в С++. Передача параметров в функции по значению и по ссылке
21. Условные конструкции в С++. Вложенные конструкции. Оператор выбора в С++

7.2. 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 сформированности знаний, умений и навыков обучающегося, необходимым для освоения дисциплины и приобретенным в результате освоения предшествующих дисциплин.

2. Тематический контроль определяет 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 тестирование. Изучение дисциплины завершается контрольной работой, проводимой в виде письменного опроса с учетом текущего рейтинга.
Неявка студента на промежуточный контроль в установленный срок без уважительной причины оценивается нулевым баллом. Повторная сдача в течение семестра не разрешается.
Дополнительные дни отчетности для студентов, пропустивших контрольную работу по уважительной причине, подтвержденной документально, устанавливаются преподавателем дополнительно.
Лабораторные занятия, пропущенные без уважительной причины, должны быть отработаны до следующей контрольной точки, если сдаются позже, то оцениваются в 1 балл.
Итоговой формой контроля знаний, умений и навыков по дисциплине является **Экзамен.** Экзамен проводится в форме тестирования. При соответствии ответа учащегося на экзамене более чем 51 % критериев из этого списка выставляется оценка «удовлетворительно», 66% – 85% оценка «хорошо», 86% и выше оценка «отлично».

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

а) основная литература:

1. Павловская, Т. А. C/C++. Системное программирование [Текст]: Учебное пособие / Т. А. Павловская .- СПб. :Питер , 2011. – 347с
2. Павловская, Т.А. C/C++. Программирование на языке высокого уровня [Текст]: для магистров и бакалавров. - СПб. [и др.] : Питер, 2012. - 460 с. - (Учебник для вузов).
3. Программирование на языке высокого уровня C/C++ [Электронный ресурс]: конспект лекций/ — Электрон. текстовые данные.— М.: Московский государственный строительный университет, Ай Пи Эр Медиа, ЭБС АСВ, 2016.— 140 с.— Режим доступа: http://www.iprbookshop.ru/48037.html.— ЭБС «IPRbooks»[Дата обращения 20 июня 2018]

б) дополнительная литература

1. Программирование на языке высокого уровня [Электронный ресурс]: методические указания и варианты заданий для студентов 1-го курса направления подготовки 09.03.01 Информатика и вычислительная техника/ — Электрон. текстовые данные.— М.: Московский государственный строительный университет, Ай Пи Эр Медиа, ЭБС АСВ, 2016.— 89 с.— Режим доступа: http://www.iprbookshop.ru/46060.html.— ЭБС «IPRbooks» »[Дата обращения 20 июня 2019]
2. Кубенский, А.А. Структуры и алгоритмы обработки данных: объектно-ориентированный подход и реализация на C++ [Текст]: БХВ-Петербург, 2004.- 254 с.
3. Кормен, Т.и др. Алгоритмы. Построение и анализ.алгоритмов. [Текст]: Учебник / Т Кормен. - МЦНМО, Москва ,2001.- 364 с.

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

1. eLIBRARY.Ru [Электронный ресурс]: электронная библиотека / Науч. электр. б-ка.- МОСКВА.1999. – Режим доступа: http//elibrary.ru (дата обращения 15.04.2018). – Яз. рус., англ.
2. Ахмедова З.Х. Программирование на языке C++ Moodl [Электронный ресурс]: система виртуального обучения:[база данных] / Даг.гос.универ. – Махачкала, - Доступ из сети ДГУ или, после регистрации из сети ун-та, из любой точки, имеющей доступ в интернет. – URL: http: //moodl.dgu.ru. (дата обращения 22.05.18).
3. Электронный каталог НБ ДГУ Ru [Электронный ресурс]: база данных содержит сведения о всех видах лит., поступающих в фонд НБ ДГУ / Дагестанский гос.унив. – Махачкала. – 2010. – Режим доступа: http//elib.dgu.ru. свободный (дата обращения 11.03.2018)

4. Национальный Открытый Университете «ИНТУИТ» [Электронный ресурс]:электронно-библиотечная система, издательство «Лань» - www.intuit.ru (дата обращения 12.03.2018)

10. Методические указания для обучающихся по освоению дисциплины.

К современному специалисту общество предъявляет достаточно широкий перечень требований, 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.

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 обеспечения для проведения лабораторных занятий;
2. Microsoft Visual Studio (или CodeBloc) для выполнения лабораторных заданий
3. Лекционная мультимедийная аудитория для чтения лекций с использованием мультимедийных материалов.
4. Visual J++, Visual C# Microsoft Imagine Premium.

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

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

Лекционные занятия

- Мультимедийное оборудование:

ноутбук SONYVGN- FZ31MR
Проектор BenQMP670.

Лабораторные занятия

- 26 компьютеров Intel COREL5 с выходом в сеть Интернет.