Министерство науки и высшего образования Российской Федерации

ФГБОУ ВО «Тверской государственный университет»

Утверждаю:
Руководитель ООП:
_____ Н.А. Семыкина
«9» 06 2023 г.

Рабочая программа дисциплины (с аннотацией)

Языки программирования

Специальность

10.05.01 Компьютерная безопасность

Специализация

«Математические методы защиты информации»

Для студентов 1 – 3 курсов очной формы обучения

Уровень высшего образования

СПЕЦИАЛИТЕТ

Составители:
ст. преподаватель С.А.Желтов,
ст. преподаватель Е.В. Тишина
к.ф.-м.н., доцент И.А.Шаповалова

Тверь 2023

I. Аннотация

1. Наименование дисциплины (модуля) в соответствии с учебным планом

Языки программирования

2. Цель и задачи дисциплины (модуля)

Целью освоения дисциплины (модуля) является:

формирование базы для развития профессиональных компетенций, связанных с готовностью студента к деятельности в области проектирования и разработки информационных моделей, предназначенных для решения различных профессиональных, исследовательских и прикладных задач.

Задачами освоения дисциплины (модуля) являются:

- получение базовых знаний и умений, связанных с разработкой алгоритмов и формирование алгоритмического мышления;
- получение теоретических знаний о роли и назначении различных языков программирования высокого и низкого уровня;
- обучения студентов общим принципам использования языков программирования; средствам описания данных; средствам описания действий; абстрактным типам данных;
- развитие навыков программирования на языках C/C++ с использованием современных интегрированных сред разработки (IDE) и инструментальных средств;
- получение теоретических знаний и практических навыков об архитектуре ЭВМ и синтаксисе языка Ассемблер;
- формирование навыков мышления программиста и создания ПО для решения различных профессиональных, исследовательских и прикладных задач, а также содействовать фундаментализации образования и развитию системного мышления.

3. Место дисциплины (модуля) в структуре ООП

Дисциплина входит в базовую часть профессионального цикла дисциплин и формирует компетенцию ОПК-8.

ОПК-8. способность использовать языки и системы программирования, инструментальные средства для решения различных профессиональных, исследовательских и прикладных задач.

Данная компетенция формируется на основе ОПК-7, является базовой для формирования ПК-10, ПК-12, ПСК-2.4, ПСК-2.5.

Для освоения дисциплины студент должен владеть современными методами и средствами информационных технологий. Необходимы знания, умения и компетенции, полученные студентами на занятиях по предмету информатика в средней общеобразовательной школе и необходимы компетенции, сформированные в процессе обучения дисциплины Информатика, Принципы алгоритмизации. Дисциплина “Языки программирования” является базовой для изучения дисциплин по операционным системам, системам управления базами данных, методам программирования. Знания и практические навыки, полученные из курса,

используются студентами при изучении научных дисциплин, при прохождения производственной и преддипломной практики, а также при разработке курсовых и дипломных работ.

4. Объем дисциплины (или модуля):
___**20**___ зачетных единиц, __**720**__ академических часов, **в том числе контактная работа:** лекции __144_____ часов, лабораторные работы ___252____ часов, **самостоятельная работа:** ___**324**__ часов.

5. Перечень планируемых результатов обучения по дисциплине (или модулю), соотнесенных с планируемыми результатами освоения образовательной программы

ОПК-8. способностью использовать языки и системы программирования, инструментальные средства для решения различных профессиональных, исследовательских и прикладных задач

Планируемые результаты освоения образовательной программы (формируемые компетенции)	**Планируемые результаты обучения по дисциплине (модулю)**
Базовый	**Владеть:** профессиональной терминологией в области информационной безопасности. **Уметь:** формализовать поставленную задачу; работать с интегрированными средами разработки программного обеспечения. **Знать:** общие принципы построения и использования современных языков программирования высокого уровня; язык программирования высокого уровня (объектно-ориентированное программирование); базовые структуры данных.
Продвинутый	**Владеть:** навыками разработки, документирования, тестирования и отладки программ; навыками разработки алгоритмов решения типовых профессиональных задач. **Уметь:** разрабатывать эффективные алгоритмы и программы; планировать разработку сложного программного обеспечения **Знать:** современные технологии программирования; язык ассемблера персонального компьютера; особенности взаимодействия языков высокого и низкого уровня, организации работы с памятью в скриптовых языках

6. Форма промежуточной аттестации:
зачет, экзамен

7. Язык преподавания русский**.**

II. Содержание дисциплины (или модуля), структурированное по темам (разделам) с указанием отведенного на них количества академических часов и видов учебных занятий

Для студентов очной формы обучения

Учебная программа – наименование разделов и тем	**Всего (час.)**	**Контактная работа (час.)**		**Самостоятельная работа (час.)**
		Лекции	Практические (лабораторные) занятия	
Раздел 1. Структурное программирование				
Тема 1.1. Общие принципы построения и использования языков программирования	22	2	10	10
Тема 1.2. Лексические основы с++. Типы и объявления	28	2	10	16
Тема 1.3. Операторы и инструкции с++	42	6	16	20
Тема 1.4. Указатели и массивы	50	10	20	20
Тема 1.5. Функции	42	4	18	20
Тема 1.6. Файлы	36	4	12	20
Тема 1.7. Обработка исключений	40	8	12	20
Раздел 2. Основы объектно-ориентированной парадигмы Программирования на с++				
Тема 2.1. Объектно-ориентированное программирование и с++	26	6	10	10
Тема 2.2. Класс – абстрактный тип данных	36	6	20	10
Тема 2.3. Перегрузка операторов, копирование и преобразование	50	12	20	18
Тема 2.4. Производные классы	36	10	16	10
Раздел 3. Основы обобщенной парадигмы программирования на с++	30	8	12	10
Тема 3.1. Шаблоны как поддержка обобщенного программирования	26	8	12	6
Тема 3.2. Использование параметризованных классов	18	4	8	6
Тема 3.3. Организация стандартной библиотеки с++	58	18	20	20
Раздел 4. Устройство компьютера.				

Понятие архитектуры.				
Тема 4.1. Структурная схема и архитектура компьютера.	8	5	2	1
Тема 4.2. Параллельная обработка.	8	5	2	1
Тема 4.3. Языки параллельного программирования	8	5	2	1
Тема 4.4. Общая характеристика языка ассемблера.	25	6	12	7
Тема 4.5 взаимодействие программ на языке ассемблера с ос	23	5	12	6
Тема 4.6. Способы реализации системно-зависимых программ	10	5	4	1
Тема 4.7. Макрообработка	8	5	2	1
ИТОГО	**630**	**144**	**252**	**234**

Учебная программа

Раздел 1. СТРУКТУРНОЕ ПРОГРАММИРОВАНИЕ

Тема 1.1. ОБЩИЕ ПРИНЦИПЫ ПОСТРОЕНИЯ И ИСПОЛЬЗОВАНИЯ ЯЗЫКОВ ПРОГРАММИРОВАНИЯ

Эволюция архитектуры программного обеспечения. Теоретические проблемы разработки языков программирования. История языков программирования. Общие принципы построения и использования языков программирования. Синтаксис и семантика языка программирования. Средства описания данных. Средства описания действий. Стандарты языков программирования. C++ и поддержка основных парадигм программирования – императивной, модульной, объектно-ориентированной и обобщенной. Обзор языков программирования C и C++. Структура программы на C и C++. Библиотеки программ и классов. Стандартные библиотеки C и C++. Современные интегрированные среды разработки программ. Интегрированная среда языка C++. Команды меню оболочки. C++. Установка режимов работы в среде C++. Графический интерфейс пользователя. Компиляция программы. Макрообработка программы. Препроцессор, директивы препроцессора (макросы). Использование макросов.

Тема 1.2. ЛЕКСИЧЕСКИЕ ОСНОВЫ C++. ТИПЫ И ОБЪЯВЛЕНИЯ

Алфавит и лексемы. Идентификаторы и ключевые слова. Пространство имен. Операторы и операции. Разделители. Ввод и вывод в C и C++. Математические вычисления в C и C++.

Средства описания данных. Фундаментальные типы – логический тип, символьные типы, целые типы, типы с плавающей точкой. Тип void. Типы, определяемые пользователем. Указатели. Массивы. Ссылки. Объявления и определения. Имена. Константы. Инициализация. Классы памяти, классификация переменных по классам памяти.

Тема 1.3. ОПЕРАТОРЫ И ИНСТРУКЦИИ C++

Средства описания действий. Обзор стандартных операторов. Инструкции языка C / C++. Оператор-выражение. Блоки, составной оператор. Управляющие операторы. Операторы цикла. Примеры и особенности использования. Составные

типы языка C/C++: Массивы, Структуры (struct), Объединения (union), Перечисляемые типы (enum), Битовые поля. Оператор typedef.

Тема 1.4. УКАЗАТЕЛИ И МАССИВЫ

Память. Описание данных как инструмент распределения памяти и контроля обработки данных. Указатели и адресная арифметика. Классификация указателей. Указатели на объекты. Операции над указателями. Массивы и указатели. Указатели и константы. Указатели на функции. Отладчики. Отладчик интегрированной системы C++. Строки. Строковая константа, ввод строк с клавиатуры, стандартные функции для строк. Массивы строк, массив указателей.

Тема 1.5. ФУНКЦИИ

Оформление программы в виде функции. Объявления и определения функций. Имя функции, список формальных параметров функции, тело функции, область действия функции. Вызовы функций. Аргументы функции. Передача факт параметров в функцию, передача массивов в функцию. Аргументы по умолчанию. Указатели на функцию. Рекурсивные функции. Встраивание функций. Перегрузка функций.

Тема 1.6. ФАЙЛЫ

Файлы прямого доступа, режим работы в файлах прямого доступа. Файлы последовательного доступа, операции доступа к элементам файла. Потоковые файлы, ввод-вывод в файловых потоках. Обработка файлов. Директивы форматного ввода-вывода.

Тема 1.7. ОБРАБОТКА ИСКЛЮЧЕНИЙ

Обработка исключений. Обнаружение ошибок и обработка сбоев работы программ. Операторы try, catch, throw как средство обработки исключительных ситуаций.

Раздел 2. ОСНОВЫ ОБЪЕКТНО-ОРИЕНТИРОВАННОЙ ПАРАДИГМЫ ПРОГРАММИРОВАНИЯ НА C++

Тема 2.1. ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ И C++

Архитектура системы – структуры классов и объектов системы. Объектно-ориентированное программирование. Эволюция объектной модели. Объектно-ориентированная декомпозиция. Принципы объектной модели – абстрагирование, инкапсуляция, модульность, иерархичность, типизация. Абстрактные типы данных: инкапсуляция, спецификация, реализация, параметризация, классы и объекты. Принципы реализации абстрактных типов данных – инкапсуляция, наследование и полиморфизм.

Тема 2.2. КЛАСС – АБСТРАКТНЫЙ ТИП ДАННЫХ

Класс как расширение понятия структуры. Область действия класса и доступ к членам класса. Отделение интерфейса от реализации. Управление доступом к членам класса Конструкторы, деструкторы. Спецификаторы доступа – собственный (закрытый), общедоступный (открытый) и защищенный. Компонентные данные и компонентные функции. Статические компоненты класса. Указатели на компоненты класса. Определение компонентных функций. Указатель this. Дружественные компоненты класса.

Тема 2.3. ПЕРЕГРУЗКА ОПЕРАТОРОВ, КОПИРОВАНИЕ И ПРЕОБРАЗОВАНИЕ

Общие принципы перегрузки операторов. Операторные функции. Бинарные и унарные операторы. Предопределенный смысл операторов. Операторы и типы,

определяемые пользователем. Операторы в пространствах имен. Перегрузка конструктора. Конструктор по умолчанию. Конструктор копирования. Конструктор преобразования. Операторы преобразования. Друзья класса.

Тема 2.4. ПРОИЗВОДНЫЕ КЛАССЫ

Наследование классов. Базовые и производные классы. Конструкторы производных классов. Иерархии классов и объектов. Одиночное наследование. Множественное наследование и виртуальные базовые классы. Полиморфизм времени выполнения (динамический полиморфизм). Виртуальные функции. Абстрактные классы. Иерархии классов и абстрактные классы. Вложенные и локальные классы. Интегрированная среда. Генераторы кода/приложений.

Раздел 3. ОСНОВЫ ОБОБЩЕННОЙ ПАРАДИГМЫ ПРОГРАММИРОВАНИЯ НА С++

Тема 3.1. ШАБЛОНЫ КАК ПОДДЕРЖКА ОБОБЩЕННОГО ПРОГРАММИРОВАНИЯ

Шаблоны. Определение шаблонов функций. Параметры шаблонов функций. Выведение типа параметров шаблона по типам аргументов при вызове функции. Переопределение шаблонов функций. Определение шаблонов классов. Параметры шаблонов классов. Создание объектов по шаблонам. Включение конструкторов в шаблон функции. Параметризация и наследование. Полиморфизм времени компиляции (параметрический полиморфизм).

Тема 3.2. ИСПОЛЬЗОВАНИЕ ПАРАМЕТРИЗОВАННЫХ КЛАССОВ

Ограниченные (защищенные) массивы. Очереди. Стеки. Связные списки. Бинарные деревья.

Тема 3.3. ОРГАНИЗАЦИЯ СТАНДАРТНОЙ БИБЛИОТЕКИ С++

Основные концепции – контейнеры, итераторы и алгоритмы. Фундаментальные последовательности – вектора, списки, очереди с двумя концами (деки). Обзор операций с последовательностями. Адаптеры последовательностей – стеки, очереди, очереди с приоритетом. Ассоциативные контейнеры. Обзор операций с ассоциативными контейнерами. Алгоритмы и объекты-функции. Библиотеки программ и классов. Обзор алгоритмов стандартной библиотеки. Итераторы и распределители памяти.

Раздел 4. УСТРОЙСТВО КОМПЬЮТЕРА. ПОНЯТИЕ АРХИТЕКТУРЫ. ОБЩАЯ ХАРАКТЕРИСТИКА ЯЗЫКА АССЕМБЛЕРА.

Тема 4.1. СТРУКТУРНАЯ СХЕМА И АРХИТЕКТУРА КОМПЬЮТЕРА.

Структурная схема компьютера. Понятие архитектуры. Принципы работы машины Фон Неймана. Программно – аппаратная архитектура IA – 32 процессоров Intel и «совместимых». Микроархитектуры.

Тема 4.2. ПАРАЛЛЕЛЬНАЯ ОБРАБОТКА.

Параллельная обработка. Модели программирования для систем с разделяемой, распределенной памятью. Разделение последовательных программ на параллельные нити. Ограничение на распараллеливание циклов. Синхронизация параллельных процессов. Автоматическое распараллеливание последовательных программ.

Тема 4.3. Языки параллельного программирования. Стандарты Open MP (Open Multi-Processing). Набор библиотек Boost.

Тема 4.4. ОБЩАЯ ХАРАКТЕРИСТИКА ЯЗЫКА АССЕМБЛЕРА.

Общая характеристика языков ассемблера: назначение, принципы построения и использования. Синтаксис ассемблера. Структура языка, основные группы

команд, операторы. Память и адресация. Стек. Регистры и их назначение. Средства взаимодействия с операционной системой (ОС). Языки ассемблера современных ЭВМ.

Тема 4.5 Взаимодействие программ на языке ассемблера с ОС: стандартные соглашения о связях, особенности использования аппаратных средств при взаимодействии программ с ОС. Структура программы ассемблера. Основные типы данных ассемблера. Процедуры в ассемблере, способы передачи параметров. Сложные типы данных. Взаимодействие программ на языке ассемблер с языками высокого уровня.

Тема 4.6. Способы реализации системнозависимых программ; особенности программирования в мультипрограммной и мультизадачной средах: повторно используемые и реентерабельные программы. Особенности работы в защищенных режимах; организация параллельной обработки.

Тема 4.7. МАКРООБРАБОТКА.

Макрообработка. Макросредства ассемблера, макрокоманды. Макро-генерация. Макропроцессоры: особенности макрообработки программ на языках ассемблера, способы использования макропроцессоров.

III. Перечень учебно-методического обеспечения для самостоятельной работы обучающихся по дисциплине (или модулю)

Самостоятельная работа обучающихся направлена на освоение учебного материала и развитие практических умений. Самостоятельная работа включает следующие виды самостоятельной работы студентов: работа с рекомендованной литературой и документацией; выполнение практических заданий; подготовка к контрольным тестам и итоговому зачёту.

Список практических заданий

1. Найти расстояние между двумя точками с заданными координатами (x_1 , y_1), (x_2 , y_2). На плоскости. Расстояние вычисляется по формуле.

$$l = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

2. Поменять местами содержимое переменных A и B и вывести новые значения A и B.
3. С клавиатуры вводится число. После этого на экран выводится следующее сообщение. Например если введено число 173, то сообщение будет «Следующее число после 173 это 174, а предыдущее 172»
4. Составьте программу вычисления площади треугольника по формуле Герона $S = \sqrt{p(p-a)(p-b)(p-c)}$, где полупериметр вычисляется по формуле $p = \frac{a+b+c}{2}$, если даны длины сторон треугольника
5. Даны три точки A, B, C на числовой оси. Найти длины отрезков AC и BC и их сумму.
6. Даны переменные A, B, C. Изменить их значения, переместив содержимое A в B, B в C, C в A, и вывести новые значения переменных A, B, C.
7. Составить программу вычисления: $y = \frac{\sqrt{1+|2+a|}}{3{,}28}$, где $a = \sqrt{2x+12}$ и x водится с клавиатуры.

8. Дано число a. Не используя никаких операций, кроме умножения, и никаких функций получите:

a^6 за три операции,

a^7 за четыре операции,

a^8 за три операции,

a^9 за четыре операции,

a^{10} за четыре операции,

9. Даны координаты двух противоположных вершин прямоугольника: (x_1, y_1), (x_2 , y_2). Стороны прямоугольника параллельны осям координат. Найти периметр и площадь данного прямоугольника.

10. С клавиатуры вводится двузначное число. Вывести на экран его последнюю цифру. (Разряд единиц)

11. Дано натуральное число. Найдите число десятков в его записи. (т.е. вторую цифру с права в его записи).

12. Найти сумму цифр заданного четырехзначного числа.

13. Найти площадь треугольника, две стороны которого равны *a* и *b*. А угол между ними γ. (Воспользоваться теоремой косинусов и формулой Герона).

14. Вычислить площадь кольца, ширина которого равна H, а отношение радиуса большей окружности к радиусу меньшей окружности D.

15. Дано целое число, лежащее в диапазоне 1-999. Вывести его строку описание вида «четное двузначное число», «нечетное трехзначное число», и т.д.

16. Даны три целых числа, одно из которых отлично от двух других, равных между собой. Определить порядковый номер числа, отличного от остальных.

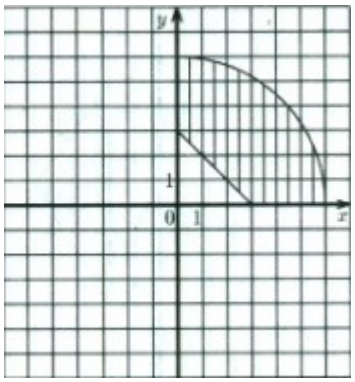
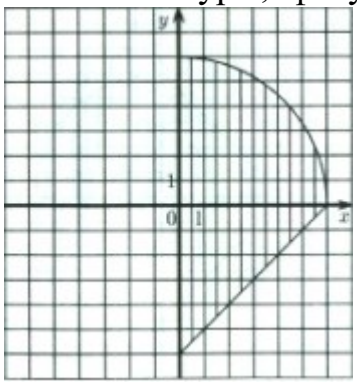
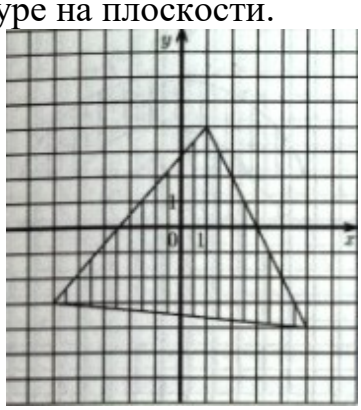
17. . Даны три целых числа. Если они упорядочены по возрастанию или убыванию, то удвоить их. В противном случае оставить без изменений. Вывести эти числа на экран.

18. Даны три числа. Найти наименьшее из них.

19. Даны три числа. Найти среднее из них (то есть число, расположенное между наименьшим и наибольшим).

20. С клавиатуры вводятся два числа A и B. Меньшее из этих чисел заменить их полусуммой, а большее удвоенным произведением

21. Составить программу определяющую, принадлежит ли точка с координатами X и Y, введенными с клавиатуры, фигуре на плоскости.

22. Найдите сумму всех двузначных чисел.

23. Дано целое число N (N>0). Используя один цикл найти сумму

1+ 1/2 + 1/3 +…+1/N

24. Дано целое число N (N>0). Найти произведение N сомножителей

1,1*1,2*1,3*…

25. Дано целое число N (N>0). Найти значение выражения из N слагаемых (знаки чередуются). Условный оператор не использовать.

1,1 - 1,2 + 1,3 - …

26. Дано вещественное число A и целое число N (N>0). Используя один цикл, вывести все целые степени числа A от 1 до N.

27. Дано вещественное число A и целое число N (N>0). Используя один цикл найти сумму:

$1 + A + A^2 + A^3 + \ldots + A^N$

28. Дано целое число N (N>0). Найти факториал числа N.

N! = 1*2*3*…*N

29. Дано целое число N (N>0). Используя один цикл найти сумму:

1! + 2! + 3! + … + N!

30. Практическая работа «Расчет конечных сумм»

вид суммы	контрольное значение
$1+2+3+4+\ldots+N$	К= $\frac{N(N+1)}{2}$
$1+3+5+7+\ldots+(2N-1)$	К= N^2
$2+4+6+8+..+2N$	К= $N(N+1)$
$1^2+2^2+3^2+4^2+\ldots+N^2$	К= $\frac{N(N+1)(2N+1)}{6}$
$1^2+3^2+5^2+\ldots+(2N-1)^2$	К= $\frac{N(4N^2-1)}{3}$
$1^3+2^3+3^3+4^3+\ldots+N^3$	К= $\frac{N^2(N+1)^2}{4}$
$1^3+3^3+5^3+\ldots+(2N-1)^3$	К= $N^2(2N^2-1)$
$1^4+2^4+3^4+4^4+\ldots+N^4$	К= $\frac{(N^2+N)(2N+1)(3N^2+3N-1)}{30}$

31. Найти все четырехзначные числа для цифр из которых состоит это число выполняется равенство АВ – СД = А+В+С+Д (Где А,В,С,Д – цифры из которых состоит число)

32. Найти все трехзначные числа, которые при делении на 2 дают остаток 1, при делении на 3 остаток 2, при делении на 4 остаток 3, а само число делится на 5. (без остатка)

33. Число Армстронга - такое число из k цифр, для которого сумма k-х степеней его цифр равна самому числу. Например, число 153 является числом Армстронга (k=3) т.к. $153=1^3+5^3+3^3$. Требуется написать программу для нахождения всех трёхзначных чисел Армстронга.

34. Написать программу, осуществляющую заполнение массива из 10 элементов случайными числами из интервала от -5 до 5 и вывода их в строчку.

35. Написать программу, заполняющую массив из 10 элементов по следующему правилу: A(i) = (2*i – 1)/sin (i);

36. Найдите 6-ой член последовательности A(i)=A(i-1)*A(i-1)+1 , если A(1)=5.

37. Заполнить массив из 10 элементов случайными числами из интервала от -10 до 10, вывести на экран, а затем увеличить каждый элемент массива на 1 и повторно вывести на экран.

38. Заполнить двумерный массив N x N случайными числами из интервала [-10 ; 10] и найти сумму элементов в каждой строке.

39. Заполнить двумерный массив N x N случайными числами из интервала [-5 ; 5] и найти произведение элементов в каждом столбце.

40. Заполнить двумерный массив N x N случайными числами из интервала [-10 ; 10] и найти сумму положительных элементов в каждой строке.

41. Элемент матрицы назовем седловой точкой, если он является наименьшим в своей строке и одновременно наибольшим в своем столбце. Или, наоборот, является наибольшим в своей строке и наименьшим в своем столбце. Для матрицы N x M введенной с клавиатуры, вывести на экран индексы всех седловых точек.

42. Заполнить двумерный массив N x N случайными числами из интервала [-10; 10], найти максимальный элемент, стоящий на главной и побочной диагоналях. Поменять его местами с элементом стоящим на пересечении этих диагоналей.

43. Заполнить двумерный массив N x N случайными числами из интервала [-10; 10], найти максимальный элемент в массиве и удалить строку, в которой находится этот элемент.

44. Заполнить двумерный массив N x N случайными числами из интервала [-10; 10],найти минимальный элемент в массиве и удалить столбец, в которой находится этот элемент.

45. С клавиатуры вводится строка символов, вывести на экран символ, стоящий на k – ом месте. k – вводится с клавиатуры.

46. С клавиатуры вводится слово, верно ли, что оно начинается и заканчивается на одну и туже букву.

47. Дано два слова. Верно ли, что первое слово заканчивается на туже букву, на которую начинается второе.

Вопросы для контрольных тестов и самоконтроля.

1. Что определяет класс? Чем обличается класс от объекта?

Класс определяет пользовательский тип: описание данных и операций над ними. Объект - это конкретный экземпляр класса со своим состоянием.

2. Можно ли объявлять массив объектов? А массив классов?

Да. Нет. Только в метапрограммировании. boost::tuple

3. Разрешается ли объявлять указатель на объект? А указатель на класс?

Да. Нет.

4. Допускается ли передавать объекты в качестве параметров, и какими способами? А возвращать как результат?

По ссылке, по указателю. Возвращать также. По значению передаётся копия объекта.

5. Как называется использование объекта одного класса в качестве поля другого класса?

Композиция.

6. Является ли структура классом? Чем класс отличается от структуры?
Да. По умолчанию в классе все поля private

7. Какие ключевые слова в C++ обозначают класс?
class, struct

8. Объясните принцип инкапсуляции.
Сокрытие деталей реализации.

9. Что такое композиция?
Использование объекта одного класса в качестве поля другого.

10. Для чего используются ключевые слова public и private?
Для определения области видимости.

11. Можно ли использовать ключевые слова public и private в структуре?
Да.

12. Существуют ли ограничения на использование public и private в классе? А в структуре?
Нет. Нет.

13. Обязательно ли делать поля класса приватными?
Нет.

14. Что такое метод? Как вызывается метод?
Метод - это функция, определённая в классе.
Если статический, то вызывается для класса, если обычный - то для объекта класса.

15. Может ли метод быть приватный?
Да.

16. Как определить метод непосредственно внутри класса? А вне класса? Чем эти определения отличаются?
class A{ void method x{}};

17. Можно в методах присваивать параметрам значения по умолчанию?
Да. Существуют дополнительные ограничения.

18. Что обозначается ключевым словом this?
Указатель на объект, который вызвал метод.

19. Зачем нужны константные методы? Чем отличается определение константного метода от обычного?
Чтобы предотвратить случайное изменение данных внутри метода и показать, что метод не меняет состояние класса.

20. Может ли константный метод вызываться для объектов-переменных? А обычный метод — для объектов-констант?
Да. Нет.

21. Объясните принцип полиморфизма.
Возможность работать с объектами разных классов одинаковым образом.
Поддержка различного поведения родственных классов, предоставляемого через единый интерфейс базового класса.
Есть статический - на этапе компиляции. Шаблоны, перегрузка функций и операторов.

Есть динамический - виртуальные функции. Точное значение операции определяется объектом для которого она вызывается.

22. Что будет напечатано в результате работы следующей программы на Си++?

```
class X {
public:
        virtual void f() {cout << "X::f\n"; g(); }
        void g() { cout << "X::g\n";}
};
class Y : public X {
public:
        void f() { cout << "Y::f\n"; }
        void g() { cout << "Y::g\n"; f();}
};
class Z : public Y {
public:
        void f() { cout << "Z::f\n"; }
        void g() { cout << "Z::g\n"; f();}
};
void P(X*px,Y*py) {
        px->f(); px->g(); py->f(); py->g();
        delete px; delete py;
}
int main() {
        P(new X, new Y);
        P(new Y, new Z);
        return 0;
}
```

Тематика курсовых работ (4 семестр)

Целями курсового проектирования являются:

- закрепление знаний, полученных в ходе теоретического и практического изучения дисциплины «Программирование (Объектно-ориентированное программирование)»;
- приобретение навыков практического программирования с использованием объектно-ориентированной парадигмы;
- изучение современных систем программирования и сред для разработки объектно-ориентированных программ и систем;
- изучение отдельных разделов предметной области, не вошедших в программу теоретического обучения, формирование навыка поиска информации по конкретной теме, ее анализа и использования для решения задачи;

Курсовая работа позволяет сформировать способности будущего специалиста к самостоятельному решению практических задач и инжнерных проблем с использованием теоретических положений, а также знаний и умений, полученных в ходе обучения объектно-ориентированному программированию.

Задачей курсового проектирования является разработка иерархии типов в заданной предметной области, включающая в себя:

- разработку модели классов на языке UML;
- разработку абстрактных классов и интерфейсов;
- разработку тестирующего приложения;
- разработку документации;
- подготовку презентации.

1. Разработка класса для представления множества целых чисел на основе связанного списка. Операции – включение элемента, исключение элемента, поиск элемента, объединение множеств, пересечение множеств.
2. Разработка класса для представления упорядоченного множества строк на основе бинарного дерева. Операции – включение элемента, исключение элемента, поиск элемента, объединение множеств, пересечение множеств.
3. Создать шаблон циклической очереди. С помощью него обработать ввод с клавиатуры, заполнение информацией вашей памяти.
4. Создать шаблон приоритетной очереди. При добавлении элемента в такую очередь порядковый номер нового элемента определяется его приоритетом..
5. С помощью шаблона класса стек написать программу калькулятор с операциями сложения, вычитания, деления, умножения, возведения в степень.
6. Построить систему классов для описания плоских геометрических фигур: круга, квадрата, прямоугольника. Предусмотреть методы для создания объектов, перемещения на плоскости, изменения размеров и вращения на заданный угол. Написать программу, демонстрирующую работу с этими классами. Программа должна содержать меню, позволяющее осуществить проверку всех методов классов.
7. Программа «Англо-русский и русско-английский словарь». «База данных» словаря должна содержать синонимичные перевода слов. Программа должна обеспечивать выбор с помощью меню и выполнение одной из следующих функций:
- Загрузка «базы данных» словаря (из файла).
- Выбор режима работы: англо-русский или русско-английский.
- Вывод перевода заданного английского слова.
- Вывод перевода заданного русского слова.
- Базу данных словаря реализовать в виде двух контейнеров типа tap.
8. Программа, реализующую игру «Крестики-нолики» между двумя игроками: пользователем и компьютером (роботом). В программе использовать контейнерные классы STL.
9. Создание игры «Змейка». Реализовать игру «Змейка» в виде апплета и разместить на сайте кафедры АСОИУ в разделе выполненных курсовых проектов. Описание: Игрок управляет длинным, тонким существом, напоминающим змею, которое ползает по плоскости (как правило, ограниченной стенками), собирая еду (или другие предметы), избегая столкновения с собственным хвостом и краями игрового поля. В некоторых вариантах на поле присутствуют дополнительные препятствия. Каждый раз, когда змея съедает кусок пищи, она становится длиннее, что постепенно усложняет игру. Игрок управляет направлением движения головы змеи (обычно 4 направления: вверх, вниз, влево, вправо), а хвост змеи движется следом. Игрок не может остановить движение змейки.
10. Игра-аркада “Snake” Правила игры: Чтобы выиграть в KSnake, вам нужно съесть все яблоки в комнате и выйти через дверь, которая откроется вверху. С каждым съеденным яблоком вы становитесь длиннее. Если вы врежетесь в стену, вы

умрете. Если вы врежетесь в себя, вы умрете. Если вам в голову попадет мяч, вы умрете. Если вы слишком долго не будете есть яблоки, появятся новые.

11. Реализовать игру «Сапёр». Описание: обычно двухмерное прямоугольное игровое поле поделено на клетки или другие части, некоторые из которых содержат скрытые мины. Игрок открывает клетки, стараясь не попасть на мину. Если игрок откроет клетку (или другую область) с миной, игра заканчивается. Если же мины нет, то в клетке появляется число, обозначающее количество мин в соседних клетках (в различных вариантах игры соседство может определяться по-разному). Рассчитав при помощи таких чисел расположение мин, игрок может пометить соответствующие клетки специальной меткой, чтобы случайно не открыть их.

12. Реализация игры «Жизнь» Конвея. Место действия этой игры, «вселенная» – это размеченная на клетки поверхность, безграничная, ограниченная, или замкнутая. В компьютерных реализациях игры чаще всего используют поверхность тора. Каждая клетка на этой поверхности может находиться в двух состояниях: быть живой или быть мёртвой. Клетка имеет восемь соседей. Распределение живых клеток в начале игры называется первым поколением. Каждое следующее поколение рассчитывается на основе предыдущего по таким правилам:

- пустая (мёртвая) клетка ровно с тремя живыми клетками-соседями оживает;
- если у живой клетки есть две или три живые соседки, то эта клетка продолжает жить; в противном случае (если соседок меньше двух или больше трёх) клетка умирает (от «одиночества» или от «перенаселённости»).
- Игрок не принимает прямого участия в игре, а лишь расставляет «живые» клетки, которые взаимодействуют согласно правилам уже без его участия.
- Начальные конфигурации формируются случайно (с помощью генератора случайных чисел для заданной вероятности). Эволюция, порождаемая заданной начальной конфигурацией состояний клеток, может быть изучена в общем случае только путём её пошагового воспроизведения.

13. Реализация игры «Инь-Ян». Игра «Инь-Ян» представляет собой клеточный автомат, состояния ячеек и правила переключения которого приближённо отражают закон единства и борьбы противоположностей. Ячейки автомата имеют три состояния: пустая ячейка (мёртвая), живая ячейка Инь и живая ячейка Ян. Соседние по миру ячейки (у каждой ячейки их 8), если они живые, называются соседями. Правила переключения определены таким образом, чтобы популяции ячеек Инь и Ян противоборствовали, но не могли развиваться друг без друга. Вот эти правила:

- Рождение. У пустой ячейки ровно три соседа (живых) и они не все одинаковые, то в ней рождается Ян, когда среди соседей только один Ян, или Инь, когда среди соседей только один Инь.
- Гибель от перенаселения (одиночества). Живая ячейка, имеющая больше четырех (меньше двух) соседей, умирает от перенаселения (от одиночества);
- Гибель в неравном противостоянии. У живой ячейки ровно четыре соседа, из которых большинство – противоположного типа – ячейка умирает.
- Начальные конфигурации формируются случайно (с помощью генератора случайных чисел для заданных вероятностей Инь и Ян). Эволюция, порождаемая заданной начальной конфигурацией состояний клеток, может быть изучена в общем случае только путём её пошагового воспроизведения. Однако для каждого значения вероятности генерации можно отследить статистику результата через n шагов и говорить в каком случае будет больше вероятность вырождения автомата за заданное количество шагов.

IV. Фонд оценочных средств для проведения промежуточной аттестации обучающихся по дисциплине (или модулю)

Типовые контрольные задания для проверки уровня сформированности компетенции ОПК-8. способность использовать языки и системы программирования, инструментальные средства для решения различных профессиональных, исследовательских и прикладных задач.

Рассматривается трехкомпонентной структура компетенции: знать, уметь, владеть.

При этом под указанными категориями понимается:

- «знать» – воспроизводить и объяснять учебный материал с требуемой степенью научной точности и полноты;
- «уметь» – решать типичные задачи на основе воспроизведения стандартных алгоритмов решения;
- «владеть» – решать усложненные задачи на основе приобретенных знаний, умений и навыков, в нетипичных ситуациях

Этап формирования компетенции, в котором участвует дисциплина	**Типовые контрольные задания для оценки знаний, умений, навыков (2-3 примера)**	**Показатели и критерии оценивания компетенции, шкала оценивания**
Базовый		
Начальный **владеть**	Составить блок-схему алгоритма.	• Имеется полное верное решение, включающее правильный ответ – 3 балла • Дано верное решение, но в решении имеются неверные записи, не отделенные от решения – 2 балла • Имеется верное решение части описания из-за логической ошибки – 1 балл • Решение не дано ИЛИ дано неверное решение – 0 баллов
	по блок-схеме определить результат	• Имеется полное верное решение, включающее правильный ответ – 3 балла • Дано верное решение, но в решении имеются неверные записи, не отделенные от решения – 2 балла • Имеется верное решение части программы из-за логической

		ошибки – 1 балл • Решение не дано ИЛИ дано неверное решение – 0 баллов
Начальный **уметь**	по словесному описанию алгоритма составлять блок-схемы	• Факты и примеры в полном объеме обосновывают выводы – 2 балла • Допущена фактическая ошибка, не приведшая к существенному искажению смысла – 1 балл • Допущены фактические и логические ошибки, свидетельствующие о непонимании темы – 0 баллов
	Определить максимальное из чисел.	• Имеется полное верное решение, включающее правильный ответ – 3 балла • Дано верное решение, но в решении имеются неверные записи, не отделенные от решения – 2 балла • Имеется верное решение части программы из-за логической ошибки – 1 балл • Решение не дано ИЛИ дано неверное решение – 0 баллов
Начальный **знать**	способы описания алгоритма.	• Факты и примеры в полном объеме обосновывают выводы – 2 балла • Допущена фактическая ошибка, не приведшая к существенному искажению смысла – 1 балл • Допущены фактические и логические ошибки, свидетельствующие о непонимании темы – 0 баллов
	Свойства алгоритма	• Факты и примеры в полном объеме обосновывают выводы – 2 балла • Допущена фактическая ошибка, не приведшая к существенному искажению смысла – 1 балл • Допущены фактические и логические ошибки, свидетельствующие о непонимании темы – 0 баллов

промежуточный **владеть**	Прокомментировать по программе использование основных инструкций языка программирования C/C++	• Факты и примеры в полном объеме обосновывают выводы – 2 балла • Допущена фактическая ошибка, не приведшая к существенному искажению смысла – 1 балл • Допущены фактические и логические ошибки, свидетельствующие о непонимании темы – 0 баллов
	Приёмами структурного программирования, навыками создания программ: линейной конструкции, ветвления, циклической.	• Факты и примеры в полном объеме обосновывают выводы – 2 балла • Допущена фактическая ошибка, не приведшая к существенному искажению смысла – 1 балл • Допущены фактические и логические ошибки, свидетельствующие о непонимании темы – 0 баллов
промежуточный **уметь**	Реализуйте программу для работы с одномерными и двумерными массивами	• Имеется полное верное решение, включающее правильный ответ – 3 балла • Дано верное решение, но в решении имеются неверные записи, не отделенные от решения – 2 балла • Имеется верное решение части программы из-за алгоритмической ошибки – 1 балл • Решение не дано ИЛИ дано неверное решение – 0 баллов
промежуточный **знать**	Тпиы данных языка C/C++	• Факты и примеры в полном объеме обосновывают выводы – 2 балла

		• Допущена фактическая ошибка, не приведшая к существенному искажению смысла – 1 балл • Допущены фактические и логические ошибки, свидетельствующие о непонимании темы – 0 баллов
	Инсрукции языка С/С++	• Факты и примеры в полном объеме обосновывают выводы – 2 балла • Допущена фактическая ошибка, не приведшая к существенному искажению смысла – 1 балл Допущены фактические и логические ошибки, свидетельствующие о непонимании темы – 0 баллов
заключительный **владеть**	Навыками применения указателей с программах	• Имеется полное верное решение, включающее правильный ответ – 3 балла • Дано верное решение, но в решении имеются неверные записи, не отделенные от решения – 2 балла • Имеется верное решение части модели из-за логической ошибки – 1 балл • Решение не дано ИЛИ дано неверное решение – 0 баллов
	Приёмами процедурного программирования.	• Имеется полное верное решение, включающее правильный ответ – 3 балла • Дано верное решение, но в решении имеются неверные записи, не отделенные от решения – 2 балла • Имеется верное решение части программы из-за алгоритмической ошибки – 1 балл • Решение не дано ИЛИ дано неверное решение – 0 баллов

заключительный **уметь**	работать с одномерными и двумерными массивами	• Имеется полное верное решение, включающее правильный ответ – 3 балла • Дано верное решение, но в решении имеются неверные записи, не отделенные от решения – 2 балла • Имеется верное решение части программы из-за логической ошибки – 1 балл • Решение не дано ИЛИ дано неверное решение – 0 баллов
	Заполнить двумерный массив N x N случайными числами из интервала [-10; 10] и найти максимальный и минимальный элемент в каждой строке и поменять их местами.	• Имеется полное верное решение, включающее правильный ответ – 3 балла • Дано верное решение, но в решении имеются неверные записи, не отделенные от решения – 2 балла • Имеется верное решение части программы из-за алгоритмической ошибки – 1 балл • Решение не дано ИЛИ дано неверное решение – 0 баллов
заключительный **знать**	Стандартные алгоритмы над массивами	• Факты и примеры в полном объеме обосновывают выводы – 2 балла • Допущена фактическая ошибка, не приведшая к существенному искажению смысла – 1 балл • Допущены фактические и логические ошибки, свидетельствующие о непонимании темы – 0 баллов
	Способы сортировки массива	• Факты и примеры в полном объеме обосновывают выводы – 2 балла • Допущена фактическая ошибка, не приведшая к существенному искажению смысла – 1 балл • Допущены фактические и логические ошибки, свидетельствующие о непонимании темы – 0 баллов

Этап формирования компетенции, в котором участвует дисциплина	**Типовые контрольные задания для оценки знаний, умений, навыков (2-3 примера)**	**Показатели и критерии оценивания компетенции, шкала оценивания**
Продвинутый		
Начальный **владеть**	навыками структурного и процедурного программирования.	• Имеется полное верное решение, включающее правильный ответ – 3 балла • Дано верное решение, но в решении имеются неверные записи, не отделенные от решения – 2 балла • Имеется верное решение части описания из-за логической ошибки – 1 балл • Решение не дано ИЛИ дано неверное решение – 0 баллов
	навыками написания алгоритмов решения типовых профессиональных задач	• Имеется полное верное решение, включающее правильный ответ – 3 балла • Дано верное решение, но в решении имеются неверные записи, не отделенные от решения – 2 балла • Имеется верное решение части программы из-за логической ошибки – 1 балл • Решение не дано ИЛИ дано неверное решение – 0 баллов
Начальный **уметь**	выполнять отладку программного кода с IDE	• Факты и примеры в полном объеме обосновывают выводы – 2 балла • Допущена фактическая ошибка, не приведшая к существенному искажению смысла – 1 балл • Допущены фактические и логические ошибки, свидетельствующие о непонимании темы – 0 баллов

	формализовать поставленную задачу;	• Факты и примеры в полном объеме обосновывают выводы – 2 балла • Допущена фактическая ошибка, не приведшая к существенному искажению смысла – 1 балл • Допущены фактические и логические ошибки, свидетельствующие о непонимании темы – 0 баллов
Начальный **знать**	Способы использования динамической памяти	• Факты и примеры в полном объеме обосновывают выводы – 2 балла • Допущена фактическая ошибка, не приведшая к существенному искажению смысла – 1 балл • Допущены фактические и логические ошибки, свидетельствующие о непонимании темы – 0 баллов
	Что означает перегрузка функции?	• Факты и примеры в полном объеме обосновывают выводы – 2 балла • Допущена фактическая ошибка, не приведшая к существенному искажению смысла – 1 балл • Допущены фактические и логические ошибки, свидетельствующие о непонимании темы – 0 баллов
промежуточный **владеть**	Описать класс Circle, соответствующую окружностям. Окружность задаётся координатами центра и радиусом. Определить функцию, проверяющую, пересекаются ли две окружности. Написать программу, использующую эту функцию. Предусмотреть обработку исключительных ситуаций.	• Факты и примеры в полном объеме обосновывают выводы – 2 балла • Допущена фактическая ошибка, не приведшая к существенному искажению смысла – 1 балл • Допущены фактические и логические ошибки, свидетельствующие о непонимании темы – 0 баллов

	Объясните смысл конструкции virtual	• Факты и примеры в полном объеме обосновывают выводы – 2 балла • Допущена фактическая ошибка, не приведшая к существенному искажению смысла – 1 балл • Допущены фактические и логические ошибки, свидетельствующие о непонимании темы – 0 баллов
промежуточный **уметь**	использовать библиотеку STL	• Имеется полное верное решение, включающее правильный ответ – 3 балла • Дано верное решение, но в решении имеются неверные записи, не отделенные от решения – 2 балла • Имеется верное решение части программы из-за алгоритмической ошибки – 1 балл • Решение не дано ИЛИ дано неверное решение – 0 баллов
промежуточный **знать**	Основные понятия объектно-ориентированного программирования	• Факты и примеры в полном объеме обосновывают выводы – 2 балла • Допущена фактическая ошибка, не приведшая к существенному искажению смысла – 1 балл • Допущены фактические и логические ошибки, свидетельствующие о непонимании темы – 0 баллов
	Создайте рекурсивную функцию расчета факториального значения	• Имеется полное верное решение, включающее правильный ответ – 3 балла • Дано верное решение, но в решении имеются неверные записи, не отделенные от решения – 2 балла • Имеется верное решение части программы из-за алгоритмической ошибки – 1 балл • Решение не дано ИЛИ дано неверное решение – 0 баллов

заключительный **владеть**	Приёмами системного программирования.	• Факты и примеры в полном объеме обосновывают выводы – 2 балла • Допущена фактическая ошибка, не приведшая к существенному искажению смысла – 1 балл Допущены фактические и логические ошибки, свидетельствующие о непонимании темы – 0 баллов
	Программными пакетами MASM и TASM: знать общие функции и различия.	• Факты и примеры в полном объеме обосновывают выводы – 2 балла • Допущена фактическая ошибка, не приведшая к существенному искажению смысла – 1 балл Допущены фактические и логические ошибки, свидетельствующие о непонимании темы – 0 баллов
заключительный **уметь**	ASM. Напишите и выполните программу, которая определяет размер памяти компьютера (INT 12H), преобразует полученное значение в ASCII-формат и выводит результат на экран в следующем виде: Размер памяти nnn байтов.	• Имеется полное верное решение, включающее правильный ответ – 3 балла • Дано верное решение, но в решении имеются неверные записи, не отделенные от решения – 2 балла • Имеется верное решение части программы из-за логической ошибки – 1 балл • Решение не дано ИЛИ дано неверное решение – 0 баллов
	Объясните действие следующих команд: STC MOV BX, DATAX ADC BX, DATAY	• Факты и примеры в полном объеме обосновывают выводы – 2 балла • Допущена фактическая ошибка, не приведшая к существенному искажению смысла – 1 балл Допущены фактические и логические ошибки, свидетельствующие о непонимании темы – 0 баллов

заключительный **знать**	Знать способы организации памяти Структура и форматы команд МП Intel 80x86. Команды пересылки данных.	• Факты и примеры в полном объеме обосновывают выводы – 2 балла • Допущена фактическая ошибка, не приведшая к существенному искажению смысла – 1 балл • Допущены фактические и логические ошибки, свидетельствующие о непонимании темы – 0 баллов
	Структура команд МП: базовая, индексная и косвенная адресации.	• Факты и примеры в полном объеме обосновывают выводы – 2 балла • Допущена фактическая ошибка, не приведшая к существенному искажению смысла – 1 балл • Допущены фактические и логические ошибки, свидетельствующие о непонимании темы – 0 баллов

При оценивании результатов освоения дисциплины применяется «рейтинговая» технология (балльно-накопительная) система. Оценка уровня сформированности компетенций осуществляется в процессе следующих форм контроля:

1) **следящего** (проводится оценка выполнения студентами заданий в ходе аудиторных занятий). Дает возможность квалифицировать степень сформированности знаний, умений, навыков, а также их глубину и прочность. Его задача - регулярное управление учебной деятельности студентов и ее корректировка. Он позволяет получать первичную информацию о ходе и качестве усвоения учебного материала, а также стимулировать регулярную, напряженную и целенаправленную работу студентов. Данный контроль позволяет вовремя выявить пробелы в знаниях и оказать им помощь в усвоении программного материала. Данными формами контроля являются: ответы с места и у доски, проверка работ выполненных в тетради.

2) **текущего** (оценивается работа студентов вне аудиторных занятий). Текущими формами контроля являются: проверка выполнения практических работ, ответы у доски, рефераты, доклады, проверка самостоятельной работы студентов.

3) **промежуточного** (рейтинговые точки) позволяет определять качество изучения студентами учебного материала по разделам и темам. Контроль проводится два раз в семестр. С помощью периодического контроля обобщаются и усваиваются целые темы и разделы, выявляется взаимосвязями с другими разделами, предметами. Контроль охватывает студентов и всей группы и проводится в виде теста, письменных практических работ.

4) **итогового** (зачёт). Максимальная сумма рейтинговых баллов по дисциплине составляет 100 баллов. Студенту, набравшему 50 баллов и выше по итогам работе в семестре, в экзаменационной ведомости и зачетной книжке выставляется оценка «зачтено». Студент, набравший от 20 до 49 баллов включительно, сдаёт зачет в последнюю неделю семестра по данной дисциплине. Баллы, полученные на зачете проставляются в ведомости. Студенту, набравшему меньше 20 баллов, в экзаменационной ведомости выставляется оценка «незачтено». Данному студенту разрешается пересдача зачета по направлению деканата на последней неделе семестра.

Формы контроля

Занятия для студентов очной формы обучения проводятся в 1-м и во 2-м семестрах 1 курса и заканчиваются в первом семестре письменной работой и во втором – зачётом. Период времени, отведенный на обучение по данной дисциплине, планируется разделить на 4 модуля, каждый из которых заканчивается контрольной точкой. За текущую работу в первом и во втором семестре, включая контрольные точки, студент может заработать 100 баллов. Количество баллов за текущую работу выставляется в соответствии со сложностью темы и количеством заданий, выносимых для практических работ в аудитории и самостоятельных занятий.

Занятия для студентов очной формы обучения проводятся в 1-м и во 2-м семестрах 2 курса и заканчиваются в первом семестре экзаменом и во втором – зачётом. Период времени, отведенный на обучение по данной дисциплине, планируется разделить на 4 модуля, каждый из которых заканчивается контрольной точкой. За текущую работу в первом семестре, включая контрольные точки, студент может заработать 100 баллов, во втором – 60 баллов, и 40 баллов составляет максимальная оценка за экзаменационный ответ. Количество баллов за текущую работу выставляется в соответствии со сложностью темы и количеством заданий, выносимых для практических работ в аудитории и самостоятельных занятий.

Занятия для студентов очной формы обучения проводятся в 1-м семестре 3 курса и заканчиваются экзаменом. Период времени, отведенный на обучение по данной дисциплине, планируется разделить на 4 модуля, каждый из которых заканчивается контрольной точкой. За текущую работу студент может заработать 60 баллов, и 40 баллов составляет максимальная оценка за экзаменационный ответ. Количество баллов за текущую работу выставляется в соответствии со сложностью темы и количеством заданий, выносимых для практических работ в аудитории и самостоятельных занятий.

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

Для оценки уровня теоретических и практических знаний используется тест или контрольный письменный опрос. Перечень некоторых вопросов теста и практических заданий представлен ниже.

Приводится три варианта из имеющихся двадцати различных вариантов по каждой из рассматриваемых тем.

1. Циклические вычислительные процессы

Вариант 1

Вычислить и вывести на экран в виде таблицы значения функции F на интервале от X нач. до X кон. С шагом dX.

$$F = \begin{cases} \dfrac{1}{ax-b} & при\ \ x+5<0\ \ и\ \ c=0 \\ \dfrac{x-a}{x-c} & при\ \ x+5>0\ \ и\ \ c\neq 0 \\ \dfrac{x}{c} & в\ остальных\ случаях \end{cases}$$

где a, b, c – действительные числа.

Функция F должна принимать действительное значение, если выражение (Ац И Вц) ИЛИ (Вц И Сц) не равно нулю, и целое значение в противном случае. Через Ац , Вц и Сц обозначены целые части значений a, b, c операции И и ИЛИ – поразрядные. Значения a, b, c, X нач., X кон., dX ввести с клавиатуры.

Вариант 2

Вычислить и вывести на экран в виде таблицы значения функции F на интервале от X нач. до X кон. С шагом dX.

$$F = \begin{cases} ax^2+b & при\ \ x+5<0\ \ и\ \ c=0 \\ \dfrac{x-a}{x} & при\ \ x+5>0\ \ и\ \ c\neq 0 \\ \dfrac{10x}{c-4} & в\ остальных\ случаях \end{cases}$$

где a, b, c – действительные числа.

Функция F должна принимать действительное значение, если выражение (Ац ИЛИ Вц) И (Ац И Сц) не равно нулю, и целое значение в противном случае. Через Ац , Вц и Сц обозначены целые части значений a, b, c операции И и ИЛИ – поразрядные. Значения a, b, c, X нач., X кон., dX ввести с клавиатуры.

Вариант 3

Вычислить и вывести на экран в виде таблицы значения функции F на интервале от X нач. до X кон. С шагом dX.

$$F = \begin{cases} ax^2+bx+c & при\ \ a<0\ \ и\ \ c\neq 0 \\ \dfrac{-a}{x-c} & при\ \ a>0\ \ и\ \ c=0 \\ a(x+c) & в\ остальных\ случаях \end{cases}$$

где a, b, c – действительные числа.

Функция F должна принимать действительное значение, если выражение Ац И (Вц ИЛИ Сц) не равно нулю, и целое значение в противном случае. Через Ац , Вц и Сц обозначены целые части значений a, b, c операции И и ИЛИ – поразрядные. Значения a, b, c, X нач., X кон., dX ввести с клавиатуры.

2. Одномерные массивы и указатели

Размерности массивов задаются именованными константами.

Вариант 1

В одномерном массиве, состоящем из n вещественных элементов, вычислить:

1) сумму отрицательных элементов массива;

2) произведение элементов массива, расположенных между максимальным и минимальным элементами.

Упорядочить элементы массива по возрастанию.

Вариант 2

В одномерном массиве, состоящем из n вещественных элементов, вычислить:

1) сумму положительных элементов массива;

2) произведение элементов массива, расположенных между максимальным по модулю и минимальным по модулю элементами.

Упорядочить элементы массива по убыванию.

Вариант 3

В одномерном массиве, состоящем из n целых элементов, вычислить:

1) произведение элементов массива с четными номерами;

2) сумму элементов массива, расположенных между первым и последним нулевыми элементами.

Преобразовать массив таким образом, чтобы сначала располагались все положительные элементы, ж потом – все отрицательные.

3. Двумерные массивы

Вариант 1

Дана целочисленная прямоугольная матрица. Определить:

1) количество столбцов, содержащих хотя бы один нулевой элемент;

2) номер строки, в которой находится самая длинная серия одинаковых элементов.

Вариант 2

Дана целочисленная прямоугольная матрица. Определить:

2) количество строк, не содержащих ни одного нулевого элемента;

3) максимальное из чисел, встречающихся в заданной матрице более одного раза.

Вариант 3

Дана целочисленная квадратная матрица. Определить:

4) сумму элементов в тех столбцах, которые не содержат отрицательных ;

5) минимум среди сумм модулей элементов диагоналей, параллельных побочной диагонали матрицы.

4. Структуры

Вариант 1

1. Описать структуру с именем STUDENT , содержащую следующие поля:

- NAME – фамилия и инициалы;
- номер группы;
- успеваемость (массив из пяти элементов).

2. Написать программу, выполняющую следующие действия:

- ввод с клавиатуры данных в массив, состоящий из десяти структур типа STUDENT; записи должны быть упорядочены по возрастанию номера группы;
- вывод на дисплей фамилий и номеров групп для всех студентов, включенных в массив, если средний балл студента больше 4,0;
- если таких студентов нет, вывести соответствующее сообщение.

Вариант 2

1. Описать структуру с именем STUDENT , содержащую следующие поля:

- фамилия и инициалы;
- номер группы;
- успеваемость (массив из пяти элементов).

2. Написать программу, выполняющую следующие действия:

- ввод с клавиатуры данных в массив, состоящий из десяти структур типа STUDENT; записи должны быть упорядочены по возрастанию среднего балла;
- вывод на дисплей фамилий и номеров групп для всех студентов, имеющих оценки 4 и5;
- если таких студентов нет, вывести соответствующее сообщение.

Вариант 3

1. Описать структуру с именем STUDENT , содержащую следующие поля:

- фамилия и инициалы;
- номер группы;
- успеваемость (массив из пяти элементов).

2. Написать программу, выполняющую следующие действия:

- ввод с клавиатуры данных в массив, состоящий из десяти структур типа STUDENT; записи должны быть упорядочены по алфавиту;
- вывод на дисплей фамилий и номеров групп для всех студентов,имеющих хотя бы одну оценку 2;
- если таких студентов нет, вывести соответствующее сообщение.

5. Строки и файлы

Вариант 1

Написать программу, которая считывает из текстового файла три предложения и выводит их в обратном порядке.

Вариант 2

Написать программу, которая считывает текст из файла и выводит на экран только предложения, содержащие введенное с клавиатуры слово.

Вариант 3

Написать программу, которая считывает английский текст из файла и выводит на экран слова, начинающиеся с гласных букв.

6.Функции

Вариант 1

В одномерном массиве, состоящем из n вещественных элементов, вычислить:

1) сумму отрицательных элементов массива;

2) произведение элементов массива, расположенных между максимальным и минимальным элементами.

Упорядочить элементы массива по возрастанию.

Оформить каждый пункт задания в виде функции. Все необходимые данные для функции должны передаваться им в качестве параметров. Использование глобальных переменных в функциях не допускается.

Вариант 2

В одномерном массиве, состоящем из n вещественных элементов, вычислить:

1) сумму положительных элементов массива;

2) произведение элементов массива, расположенных между максимальным по модулю и минимальным по модулю элементами.

Упорядочить элементы массива по убыванию.

Оформить каждый пункт задания в виде функции. Все необходимые данные для функции должны передаваться им в качестве параметров. Использование глобальных переменных в функциях не допускается.

Вариант 3

В одномерном массиве, состоящем из n целых элементов, вычислить:

1) произведение элементов массива с четными номерами;

2) сумму элементов массива, расположенных между первым и последним нулевыми элементами.

Преобразовать массив таким образом, чтобы сначала располагались все положительные элементы, ж потом – все отрицательные.

Оформить каждый пункт задания в виде функции. Все необходимые данные для функции должны передаваться им в качестве параметров. Использование глобальных переменных в функциях не допускается.

7.Динамические структуры данных

Вариант 1

Составить программу, которая содержит динамическую информацию о наличии автобусов в автобусном парке.

Сведения о каждом автобусе включают:

- номер автобуса;
- фамилию и инициалы водителя;
- номер маршрута.
- Программа должна обеспечивать:
- начальное формирование данных обо всех автобусах в парке в виде списка;
- при выезде каждого автобуса из парка вводится номер автобуса, и программа удаляет данные об этом автобусе из списка автобусов, находящихся в парке, и записывает эти данные в список автобусов, находящихся на маршруте;
- при въезде каждого автобуса в парк вводится номер автобуса, и программа удаляет данные об этом автобусе из списка автобусов, находящихся на маршруте, и записывает эти данные в список автобусов, находящихся в парке;
- по запросу выдаются сведения об автобусах, находящихся в парке, или об автобусах, находящихся на маршруте.

Вариант 2

Составить программу, которая содержит текущую информацию о книгах в библиотеке.

Сведения о книгах включают:

- номер УДК;
- фамилию и инициалы автора;
- название;
- год издания;
- количество экземпляров данной книги в библиотеке;
- Программа должна обеспечивать:

- начальное формирование данных обо всех книгах в библиотеке в виде двоичного дерева;
- добавление данных о книгах, вновь поступающих в библиотеку;
- удаление данных о списываемых книгах;
- по запросу выдаются сведения о наличии книг в библиотеке, упорядоченные по годам издания.

Вариант 3

Составить программу, которая содержит текущую информацию о заявках на авиабилеты.

Каждая заявка включает:

- пункт назначения;
- номер рейса;
- фамилию и инициалы пассажира;
- желаемую дату вылета.
- Программа должна обеспечивать:
- хранение всех заявок в виде списка;
- добавление заявок в список;
- удаление заявок;
- вывод заявок по заданному номеру рейса и дате вылета;
- вывод всех заявок.

8. Знакомство со стандартной библиотекой STL.

vector

1. Прибавить к каждому элементу массива среднее арифметическое его положительных элементов
2. Каждый элемент массива должен быть умножен на минимальный элемент исходного массива
3. Умножить элементы массива, делящиеся на 3 без остатка, на среднее арифметическое элементов массива, делящихся на 2 без остатка
4. Разделить элементы массива на половину максимального элемента
5. Отсортировать элементы массива в порядке возрастания суммы составляющих их цифр
6. Умножить каждый отрицательный элемент массива на произведение максимального и минимального элементов исходного массива
7. Умножить каждый элемент массива на максимальный элемент исходного массива и разделить на минимальный элемент исходного массива
8. Прибавить к каждому элементу массива сумму трех минимальных элементов массива
9. Элементы, стоящие на четных позициях массива умножить на 2, а из элементов, стоящих на нечетных позициях, вычесть сумму всех неотрицательных элементов
10. Каждую четверку элементов массива (либо оставшуюся часть массива меньшей длины) отсортировать в порядке возрастания

string

1. Разработайте функцию **&** , удаляющую из строки, переданной в параметре , лишние пробелы. Лишними считаются все пробелы в начале и конце строки, а также дополнительные пробелы между словами.
2. Разработайте с ее использованием функцию, выполняющую удаление лишних пробелов из каждой входного потока символов и вывод результирующих строк в выходной поток.
3. Разработайте функцию std::string HtmlEncode(std::string const& text), выполняющую кодирование специальных символов строки text соответствующими сущностями HTML:

- “ (двойная кавычка) заменяется на "
- ‘ (апостроф) заменяется на '
- < (знак меньше) заменяется на <
- > (знак больше) заменяется на >
- & (амперсанд) заменяется на &

Разработайте на ее основе программу, выполняющую html-кодирование текста, поступающего со стандартного потока ввода, и выводящую результат в стандартный поток вывода.

4. Разработайте функцию, выполняющую декодирование html-сущностей строки, перечисленных в варианте 3, обратно в их символьное представление.

Разработайте на ее основе программу, выполняющую декодирование html сущностей текста, поступающего со стандартного потока ввода, и выводящую результат в стандартный поток вывода.

map

5. Разработайте программу-словарь, осуществляющую перевод **слов и словосочетаний**, поступающих со стандартного потока ввода, **с английского языка на русский** с использованием заданного файла словаря и выводящую результат перевода в стандартный поток вывода. Если вводимое слово или словосочетание, отсутствует в словаре, программа должна попросить пользователя ввести перевод и запомнить его, в случае, если пользователь ввел непустую строку. Для выхода из диалога с программой пользователь должен ввести строку, состоящую из трех точек. Перед выходом программа спрашивает о необходимости сохранить изменения в файле словаря, в том случае, если в словарь были добавлены фразы во время текущей сессии работы с программой. Имя файла словаря передается программе с помощью параметра командной строки.

Stack

Обратная польская запись

1. Преобразовать выражение в ОПЗ с использованием стека
2. Реализовать алгоритм вычисления выражения, записанного в ОПЗ с использованием стека

9. ООП.

Цель работы – освоение и закрепление основных навыков объектно-ориентированного программирования, связанных с проектирование спецификаций класса, его структуры и операций интерфейса с учетом свойства инкапсуляции.

Вариант 1

Описать класс, реализующий стек. Написать программу, использующую этот класс для моделирования Т-образного сортировочного узла на железной дороге. Программа должна разделять на два направления состав, состоящий из вагонов двух типов (на каждое направление формируется состав из вагонов одного типа). Предусмотреть возможность формирования состава из файла и с клавиатуры.

Вариант 2

Описать класс, реализующий бинарное дерево, обладающее возможностью добавления новых элементов, удаления существующих, поиска элемента по ключу, а также последовательного доступа ко всем элементам. Написать программу, использующую этот класс для представления англо-русского словаря. Программа должна содержать меню, позволяющее осуществить проверку всех методов класса. Предусмотреть возможность формирования словаря из файла и с клавиатуры.

Вариант 3

Построить систему классов для описания плоских геометрических фигур: круга, квадрата, прямоугольника. Предусмотреть методы для создания объектов, перемещения на плоскости, изменения размеров и вращения на заданный угол. Написать программу, демонстрирующую работу с этими классами. Программа должна содержать меню, позволяющее осуществить проверку всех методов классов.

Вариант 4

Построить описание класса, содержащего информацию о почтовом адресе организации. Предусмотреть возможность раздельного изменения составных частей адреса, создания и уничтожения объектов этого класса. Написать программу, демонстрирующую работу с этим классом. Программа должна

Вариант 5

Составить описание класса для представления комплексных чисел. Обеспечить выполнение операций сложения, вычитания и умножения комплексных чисел. Написать программу, демонстрирующую работу с этим классом. Программа должна содержать меню, позволяющее осуществить проверку всех методов класса.

Вариант 6

Составить описание класса для объектов-векторов, задаваемых координатами концов в трехмерном пространстве. Обеспечить операции сложения и вычитания векторов с получением нового вектора (суммы или разности), вычисления скалярного произведения двух векторов, длины вектора, косинуса угла между векторами. Написать программу, демонстрирующую работу с этим классом. Программа должна содержать меню, позволяющее осуществить проверку всех методов класса.

10. ООП. Наследование и полиморфизм

Вариант 7

Описать класс Circle, соответствующую окружностям. Окружность задаётся координатами центра и радиусом. Определить функцию, проверяющую, пересекаются ли две окружности. Написать программу, использующую эту функцию. Предусмотреть обработку исключительных ситуаций.

Вариант 8

Описать абстрактный класс ISolid для геометрических тел. Класс должен содержать методы: Volume и SurfaceArea, возвращающие объём и площадь поверхности соответственно. Описать классы Cube (куб) и Cylinder (цилиндр),

наследующие этот класс ISolid. Параметры тел должны задаваться при создании экземпляра. Написать функцию, принимающую тело и выводящую на экран её название, параметры, объём и площадь поверхности. Написать программу, использующую эту функцию. Построить UML-диаграмму.

Вариант 9

Вам даны классы BinaryOperation (бинарный оператор) и Number (число), которые наследуются от базового класса Expression (выражение). Ваша задача реализовать базовый класс Expression так, чтобы не было утечек памяти. Кроме этого подумайте, какие методы стоит сделать виртуальными.

Решение 2 строчки кода в первом классе

```
#include <cassert> // assert

struct Expression{
//////////////////////   (это решение ) virtual double evaluate() const = 0;
//////////////////////   (это решение )  virtual ~Expression(){}
};

struct Number : Expression {
   Number(double value)
      : value_(value)   {}
   double value() const    { return value_; }
   double evaluate() const    { return value_; }
private:
   double value_;
};

struct BinaryOperation : Expression {
   enum {
      PLUS = '+',
      MINUS = '-',
      DIV = '/',
      MUL = '*'
   };

   BinaryOperation(Expression const *left, int op, Expression const *right)
      : left_(left), op_(op), right_(right)
   { assert(left_ && right_); }

   ~BinaryOperation()    {
      delete left_;
      delete right_;
   }

   Expression const *left() const    { return left_; }
   Expression const *right() const    { return right_; }
   int operation() const    { return op_; }
   double evaluate() const    {
      double left = left_->evaluate();
      double right = right_->evaluate();
      switch (op_)       {
      case PLUS: return left + right;
      case MINUS: return left - right;
      case DIV: return left / right;
      case MUL: return left * right;
```

```
        }
        assert(0);
        return 0.0;
    }
private:
    Expression const *left_;
    Expression const *right_;
    int op_;
};
```

Добавьте к иерархии из предыдущего упражнения класс FunctionCall. FunctionCall должен представлять вызов одной из двух предопределенных математических функций: sqrt — извлечение квадратного корня и abs — вычисление модуля числа. Функция идентифицируется строкой, переданной в качестве параметра в конструктор.

Решение - добавлено: деструктор, и функция double evaluate() const

```
#include <string> // std::string
#include <cassert>
#include <cmath>  // sqrt и fabs
// эти классы определять заново не нужно
struct Expression;
struct BinaryOperation;
struct Number;
struct FunctionCall : Expression{
    /**
     * @name имя функции, возможные варианты
     *      "sqrt" и "abs".
     *
     *      Объекты, std::string можно
     *      сравнивать с C-строками используя
     *      обычный синтаксис ==.
     *
     * @arg  выражение аргумент функции
     */
    FunctionCall(std::string const &name, Expression const *arg)      : name_(name), arg_(arg)    {
        assert(arg_);
        assert(name_ == "sqrt" || name_ == "abs");
    }

    // реализуйте оставшиеся методы из
    // интерфейса Expression и не забудьте
    // удалить arg_, как это сделано в классе
    // BinaryOperation

   double evaluate() const    {
          if (name_ == "sqrt")
                return sqrt(arg_->evaluate());
          else if ( name_ == "abs")
                return fabs(arg_->evaluate());
          assert(0);
          return 0.0;
    }
    ~FunctionCall (){    delete arg_;    }
    std::string const & name() const    {       return name_;    }
    Expression const *arg() const    {       return arg_;    }
```

```
    // and here
private:
    std::string const name_;
    Expression const *arg_;
};
```

Список вопросов к зачету 1 курс 2 семестр

1. Типы алгоритмов. Способы описания алгоритмов.
2. Базовые операции и базовые конструкции.
3. Структурное программирование.
4. Нисходящее и восходящее проектирование алгоритмов.
5. Структура программы на языке Си.
6. Выражения в языке Си.
7. Определение алгоритма и программы.
8. Простые типы данных языка Си.
9. Преобразование типов в языке Си.
10. Описание массивов на языке Си.
11. Описание функций на языке Си.
12. Стандартные функции для случайных генерации чисел.
13. Стандартные функции языка Си, предназначенные для обработки строк.
14. Стандартные функции языка Си для определения текущего времени и даты.
15. Составные типы данных в языке Си.
16. Указатели в языке Си.
17. Структуры данных в языке Си.
18. Производные типы: ссылки и указатели.
19. Виды указателей. Инициализация указателей. Адресная арифметика. Операции с указателями.
20. Динамическое распределение памяти. Функции malloc () и free ().
21. Динамическое распределение памяти.
22. Выделение и освобождение динамической памяти. Операции new и delete.
23. Утечка памяти
24. Примеры работы с массивами. Динамические массивы.
25. Связь массивов с указателями. Различия между массивами и указателями. Многомерные массивы. Динамические многомерные массивы.
26. Ввод и вывод массивов.
27. Основные приемы и методы обработки массивов.
28. Способы поиска данных в массиве.
29. Способы сортировки данных в массиве.
30. Массивы указателей, указатели на указатели, массив указателей на функции, сложные объявления.
31. Указатель на функцию. Callback (функция обратного вызова) в программировании. Примеры использования в C/C++.
32. Строки в стиле С. Понятие строки. Инициализация строк. Функции для работы со строками.
33. Понятие функции. Средства организации подпрограмм и функций в языке Си. Локальные и статические переменные функций. Аргументы

по умолчанию. Особенности передачи параметров в функцию. Вызов функций по значению. Вызов функций по ссылке и по указателю. Возврат из функций.

34. Перегрузка функций..
35. Понятие потока и файла. Двоичные и текстовые потоки. Работа с потоками средствами языка С. Функции для работы с файлами.
36. Понятие потока и файла. Двоичные и текстовые потоки. Работа с потоками средствами языка С++. Функции для работы с файлами.
37. Понятие и работа со списком. Однонаправленный и двунаправленный списки.
38. Понятие и работа с двоичным деревом.
39. Понятие и работа со стеком.
40. Понятие и работа с очередью

Вопросы для подготовки к промежуточной аттестации:

1. Что такое класс, объекты?
2. Дайте определение атрибуту (свойству) и поведению объекта.
3. Опишите методы и атрибуты класса Карта, представляющего собой карту в карточных играх.
4. Дайте определение наследованию.
5. Опишите преимущества наследования.
6. Какой из этих классов является базовым: Транспортное средство или Микроавтобус?
7. Какие из следующих классов связаны отношением наследования: классы Двигатель и Дизель или классы Двигатель и Автомобиль?
8. Что такое конструктор и деструктор?
9. Дайте определение инкапсуляции.
10. Как можно обратиться к членам объекта?
11. Дайте определение полиморфизма.
12. Дайте определение перегрузки.
13. Дайте определение связывания.
14. Дайте определение полиморфизма времени выполнения.
15. Что такое виртуальная функция?
16. Как описывается виртуальный метод в С++?
17. Дайте определение простого и множественного наследования.
18. Дайте определение абстракции.
19. В чем заключается аналогия между понятиями, используемыми при объектно-ориентированном подходе к программированию, и существительными и глаголами?
20. Какой синтаксис используется в С++ для объявления класса, производного от базового?
21. Полиморфизм времени выполнения (динамический полиморфизм).
22. Виртуальные функции. Абстрактные классы. Иерархии классов и абстрактные классы.
23. Шаблоны. Определение шаблонов функций.
24. Полиморфизм времени компиляции (параметрический полиморфизм).
25. Основные концепции – контейнеры, итераторы и алгоритмы.

26. Обзор алгоритмов стандартной библиотеки. Итераторы и распределители памяти.
27. Какие две роли выполняет наследование.
28. Какие виды наследования возможны в C++.
29. Чем отличается модификатор доступа protected от модификаторов private и public.
30. Чем открытое наследование отличается от закрытого и защищенного.
31. Какие функции не наследуются.
32. Сформулируйте правила написания конструкторов в производном классе.
33. Каков порядок вызова конструкторов. А деструкторов.
34. Можно ли в производном классе объявлять новые поля. методы.
35. Если имя нового поля совпадает с именем унаследованного, то каким образом разрешить конфликт имен.
36. Что происходит, если имя метода-наследника совпадает с именем базового метода.
37. Когда выполняется понижающее приведение типов.
38. Объясните, зачем нужны виртуальные функции.
39. Чем "раннее" связывание отличается от "позднего"?
40. Какие два вида полиморфизма реализованы в C++?
41. Дайте определение полиморфного класса.
42. Наследуются ли виртуальные функции?
43. Каковы особенности вызова виртуальных функций в конструкторах и деструкторах?
44. Можно ли сделать виртуальной перегруженную операцию, например, сложение?
45. Может ли конструктор быть виртуальным? А деструктор?
46. Создайте функцию способную принимать аргументы, способную возвращать значения.
47. Создайте рекурсивную функцию расчета факториального значения
48. Как виртуальные функции влияют на размер класса?
49. Как объявляется "чистая" виртуальная функция?
50. Дайте определение абстрактного класса.
51. Наследуются ли чистые виртуальные функции?
52. Можно ли объявить деструктор чисто виртуальным?
53. Чем отличается чистый виртуальный деструктор от чистой виртуальной функции?
54. Зачем требуется определение чистого виртуального деструктора?
55. Наследуется ли определение чистой виртуальной функции?
56. Объясните, чем отличается множественное наследование от простого?

Список вопросов к зачету 2 курс 4 семестр

1. STL - Контейнеры STL (вектор, двусвязный список, множество, карта (отображение), двусторонняя очередь) Итераторы. Алгоритмы STL.

2. Основные концепции ООП. Понятия класса и объекта. Абстракция. Инкапсуляция. Наследование. Полиморфизм. Методы, данные и свойства. Ограничение доступа к полям классам.
3. Классы. Методы и поля (данные) классов. Объявление класса в C++.
4. Разграничение доступа к полям и методам класса (спецификаторы доступа). Интерфейс и реализация класса.
5. Классы: функции-члены, конструкторы и деструкторы. Списки инициализации.
6. Конструкторы и деструкторы. Перегрузка конструкторов. (Конструктор по умолчанию. Конструктор копирования. Конструктор с аргументами)
7. Классы: Указатель this. Друзья класса. Дружественные классы и функции. Статические данные и методы. Особенности. Область применения. Пример.
8. Вложенные классы. Композиция.
9. Шаблоны класса: определение и инстанциирование.
10. Классы: перегрузка функций. перегрузка операторов.
11. Перегрузка бинарных операторов
12. Перегрузка операторов отношения и логических операторов
13. Перегрузка унарных операторов Перегрузка оператора присваивания и индекса массива
14. Классы: Принцип наследования в ООП. Варианты наследования. Публичное наследование. Защищенное наследование. Закрытое наследование. Иерархия классов
15. Классы: виртуальные функции и абстрактные классы.
16. Обобщенное программирование. Шаблоны функций. Шаблонные операторы. Шаблоны классов. Параметры шаблонов, не являющиеся типами.
17. Полиморфизм, его основные проявления, механизмы использования.
18. Понятие раннего и позднего связывания.
19. Использование виртуального механизма для реализации принципа полиморфизма.

ВОПРОСЫ К ЭКЗАМЕНУ 2 курс 3 семестр

1 вопрос

1. Парадигмы программирования, их особенности. Структурное программирование. Процедурное программирование. Модульное программирование. Объектно-ориентированное программирование. Функциональное программирование. Логическое программирование. Обобщенное программирование.
2. Типы данных в языке C++. Размеры данных. Примеры. Переменные. Константы. Примеры. Явное и неявное преобразование типов. Объявление переменных и констант в языке Си. Область видимости переменных и время жизни объектов. Локальные и глобальные переменные. Переменные, классы памяти. Статическое, автоматическое и динамическое размещение данных. Особенности и различия.
3. Унарные, бинарные, тернарная операции в языке C++. Краткая характеристика. Операторы. приоритет и очередность вычислений (из таблицы). Примеры

4. Вывод на экран и ввод с клавиатуры в языке С. Вывод на экран и ввод с клавиатуры в языке С++. Ввод и вывод данных (использование функций и потоковых классов). Примеры

5. Типы данных Структуры в С++: объявление, инициализация, доступ к полям, функции-члены структуры. Структуры. Работа со структурами. Примеры

6. Производные типы: ссылки и указатели. Ссылочный тип в С++. Понятие указателя. Виды указателей. Инициализация указателей. Адресная арифметика. Операции с указателями. Примеры.

7. Динамическое распределение памяти. Функции malloc () и free ().

8. Динамическое распределение памяти. Выделение и освобождение динамической памяти. Операции new и delete. Утечка памяти

9. Массивы. Примеры работы с массивами. Динамические массивы. Связь массивов с указателями. Различия между массивами и указателями. Многомерные массивы. Динамические многомерные массивы. Примеры.

10. Массивы, массивы указателей, указатели на указатели, массив указателей на функции, сложные объявления.

11. Строки в стиле С. Понятие строки. Инициализация строк. Функции для работы со строками. Примеры.

12. Понятие функции. Средства организации подпрограмм и функций в языке Си. Локальные и статические переменные функций. Аргументы по умолчанию. Особенности передачи параметров в функцию. Вызов функций по значению. Вызов функций по ссылке и по указателю. Возврат из функций. Перегрузка функций. Примеры.

13. Понятие потока и файла. Двоичные и текстовые потоки. Работа с потоками средствами языка С. Функции для работы с файлами.

14. Понятие потока и файла. Двоичные и текстовые потоки. Работа с потоками средствами языка С++. Функции для работы с файлами.

15. Понятие и работа со списком. Однонаправленный и двунаправленный списки.

16. Понятие и работа с двоичным деревом.

17. Понятие и работа со стеком.

18. Понятие и работа с очередью

19. Указатель на функцию. Callback (функция обратного вызова) в программировании. Примеры использования в C/C++.

20. Стандартная библиотека шаблонов STL: основные концепции. STL: последовательные и ассоциативные контейнеры. Итераторы. Алгоритмы.

21. STL - Контейнеры STL вектор.

22. STL - Контейнеры STL двусвязный список.

23. STL - Контейнеры STL множество, карта.

24. STL - Контейнеры STL string.

25. STL - Контейнеры STL дек.

26. STL - Контейнеры STL. Стек.

27. STL - Контейнеры STL. Очередь. Очередь с приоритетом.

28. STL - Итераторы. Алгоритмы STL.

29. Основные концепции ООП. Понятия класса и объекта. Абстракция. Инкапсуляция. Наследование. Полиморфизм. Методы, данные и свойства. Ограничение доступа к полям классам.

30. Классы. Методы и поля (данные) классов. Объявление класса в С++.

31. Разграничение доступа к полям и методам класса (спецификаторы доступа). Интерфейс и реализация класса.

32. Классы: функции-члены. Конструкторы и деструкторы. Перегрузка конструкторов. (Конструктор по умолчанию. Конструктор копирования. Конструктор с аргументами)

33. Классы: Указатель this.

34. Друзья класса. Дружественные классы и функции.

35. Статические данные и методы. Особенности. Область применения. Пример.

36. Взаимодействия между объектами классов. Композиция. Агрегация. Ассоциация.

2 вопрос

1. Обработка одномерных и многомерных массивов (память под массивы выделять динамически, ввод, вывод и обработку элементов массива реализовать в виде функции, в функцию передавать указатель на массив).

a. Динамические массивы: создание, работа и удаление одномерного динамического массива.

b. Динамические массивы: создание, работа и удаление многомерного динамического массива.

c. Удаление элементов из динамического массива и вставка элементов в массив.

2. Обработка символьных данных (строк) – найти что-нибудь в строке, преобразовать строку, получить новую строку из исходной и т.п. Для обработки строк реализовать функцию, в функцию передать указатель на строку.

3. Работа со структурами (Объявить структуру, создать массив элементов типа структура, заполнить его данными, найти что-нибудь в этом массиве). Все делать через функции.

4. Работа со структурами (Объявить структуру, создать массив элементов типа структура, заполнить его данными, изменить порядок следования элементов, отсортировать, выбрать некоторые записи по условию). Все делать через функции.

5. Работа со структурами (Объявить структуру, создать массив элементов типа структура, заполнить его данными, найти что-нибудь в этом массиве, изменить порядок следования элементов, отсортировать, выбрать некоторые записи по условию). Все делать через функции.

6. Работа со списком. Примеры использования.

7. Работа с двоичным деревом. Примеры.

8. Работа со стеком

a. Организация стека.

b. Примеры использования стека.

9. для 24 группы:

a. STL вектор. Примеры использования

b. STL двусвязный список. Примеры использования

c. STL дек. Примеры использования

d. STL множество. Примеры использования

e. STL карта. Примеры использования

f. STL string. Примеры использования

g. STL Стек. Примеры использования

h. STL Очередь. Очередь с приоритетом. Примеры использования

10. для 25 группы. (Программы предлагается реализовать на ЯП Python):

a. Функция вычисления факториала.

b. Сокеты. Пример простого синхронного сервера.

c. Сокеты. Пример простого синхронного клиента.

d. Строки и JSON. Пример токенизации строки по разделителю и запись токенов в JSON.

e. Хэш-функции. Пример вычисления хэша строки и записи пар (хэш, строка) в словарь (dict)

f. Псевдослучайные числа. Реализация линейного конгруэнтного генератора псевдослучайных чисел.

g. JSON и файлы. Пример записи произвольных данных в JSON и сохранение JSON в файл.

11. Работа с файлами: файловые потоки, обработка текстовых файлов. Примеры

12. Работа с файлами: файловые потоки, обработка двоичных файлов. Примеры

13. ООП. Классы

a. Реализуйте класс String для работы со строками символов.

b. Спроектируйте и реализуйте класс дата.

c. Спроектируйте и реализуйте класс complex..

d. Спроектируйте и реализуйте класс rational.

14. Тесты

ВОПРОСЫ К ЭКЗАМЕНУ 5 СЕМЕСТР

1. Место языков ассемблера среди языков программирования.
2. Структура МП Intel 80x86: используемые регистры.
3. Структура МП Intel 80x86: операционное устройство и шинный интерфейс.
4. Размещение данных в памяти. Сегментация памяти.
5. Структура регистра флагов. Команды установки флагов.
6. Структура и форматы команд МП Intel 80x86. Команды пересылки данных.
7. Способы адресации в командах МП Intel 80x86.
8. Представление данных в IBM PC: целые числа.
9. Представление данных в IBM PC: двоично-десятичные числа.
10. Представление данных в IBM PC: алфавитно-цифровые данные.
11. Представление данных в IBM PC: вещественные данные.
12. Система команд МП: команды сложения и вычитания.
13. Команды умножения и деления чисел с ФТ.
14. Структура команд МП: базовая, индексная и косвенная адресации.
15. Логические команды обработки битов.
16. Команды сдвигов и их использование.
17. Команды передачи управления: безусловные переходы. Адресация в переходах.
18. Команды передачи управления: условные переходы.
19. Команды передачи управления: организация циклов.
20. Стек. Команды работы со стеком.

21. Элементарные конструкции языка ассемблера: алфавит, ключевые слова.
22. Элементарные конструкции языка ассемблера: числа, символьные данные.
23. Элементарные конструкции языка ассемблера: имена, метки.
24. Элементарные конструкции языка ассемблера: выражения и их использование.
25. Предложения языка ассемблера: комментарии.
26. Предложения языка ассемблера: команды.
27. Предложения языка ассемблера: директивы.
28. Структура файла ассемблер-программы. Директивы оформления программы.
29. Структура файла ассемблер-программы: односегментные и многосегментные файлы.
30. Использование прерываний в ассемблер-программах.
31. Операторы в командах языка ассемблера.
32. Блочная структура программы: правила описания и вызова процедур.
33. Блочная структура программы: расположение процедур в исходном файле.
34. Блочная структура программы: внутренние и внешние процедуры.
35. Способы передачи параметров между процедурой и вызывающей программой.
36. Передача параметров между процедурой и вызывающей программой. Проблема сохранения регистров.
37. Программные пакеты MASM и TASM: этапы обработки задания (подготовка исходного файла и его трансляция).
38. Программные пакеты MASM и TASM: этапы обработки задания (компоновка объектного модуля и отладка программы).
39. Программные пакеты MASM и TASM: общие функции и различия.
40. Модели памяти и их использование в TACM.
41. Макросы: макроопределения и их использование.
42. Макросы: использование параметров и комментарии.

ДОПОЛНИТЕЛЬНЫЕ ВОПРОСЫ

1. Регистры общего назначения.
2. Регистры-указатели.
3. Регистр флагов.
4. Организация оперативной памяти.
5. Директива SEGMENT
6. Директива PROC.
7. Директива ENDP.
8. Директива ASSUME.
9. Директива END0
10. Директива указания типов.
11. Директива упрощения описания сегментов.
12. Непосредственные операнды.
13. Адресация операндов команд.
14. Команда пересылки MOV.
15. Команда работы со стеком PUSH, POP.
16. Команда загрузки адреса LEA.

17. Команды пересылки флагов.
18. Операция сложения ADD.
19. Операция сложения с переносом ADC.
20. Операция увеличения на единицу INC.
21. Операции вычитания SUB, SBB, DEC.
22. Операция изменения знака NEG.
23. Операция сравнения CMP.
24. Операции умножения MUL, IMUL.
25. Операции делениюDIV, IDIV.
26. Операция работы с битами.
27. Логические операции.
28. Операции сдвига и циклического сдвига.
29. Операции безусловного перехода.
30. Операции условной передачи информации.

V. Перечень основной и дополнительной учебной литературы, необходимой для освоения дисциплины (или модуля)

а) Основная литература

1. Зоткин, С. П. Программирование на языке высокого уровня C/C++ : учебное пособие / С. П. Зоткин. — 3-е изд. — Москва : МИСИ – МГСУ, 2018. — 140 с. — ISBN 978-5-7264-1810-0. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/108512

2. Конова, Е. А. Алгоритмы и программы. Язык C++ : учебное пособие / Е. А. Конова, Г. А. Поллак. — 5-е изд., стер. — Санкт-Петербург : Лань, 2020. — 384 с. — ISBN 978-5-8114-5431-0. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/140730

3. Барков, И. А. Объектно-ориентированное программирование : учебник / И. А. Барков. — Санкт-Петербург : Лань, 2019. — 700 с. — ISBN 978-5-8114-3586-9. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/119661

4. Беляков С. Л. Основы разработки программ на языке С++ для систем информационной безопасности : учебное пособие / С. Л. Беляков, А. В. Боженюк, М. В. Петряева; Южный федеральный университет. - Ростов-на-Дону : Издательство Южного федерального университета (ЮФУ), 2020. - 152 с. - ВО - Бакалавриат. – Режим доступа: https://znanium.com/catalog/document?id=374992

б) Дополнительная литература

1. Фридман, А. Л. Язык программирования Си++ : [16+] / А. Л. Фридман. – 2-е изд., исправ. – Москва : Национальный Открытый Университет «ИНТУИТ», 2016. – 219 с. – (Основы информационных технологий). – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=578114

2. Языки программирования: учебное пособие / О.Л. Голицына, Т.Л. Партыка, И.И. Попов. - 2-е изд., перераб. и доп. - М.: Форум, 2010. - 400 с.: ил.; 60x90 1/16. - (Профессиональное образование). (переплет) ISBN 978-5-91134-442-9. Режим доступа: http://znanium.com/go.php?id=226043

3. Кауфман В.Ш. Языки программирования. Концепции и принципы [Электронный ресурс] / В.Ш. Кауфман. — Электрон. текстовые данные. — Саратов: Профобразование, 2017. — 464 c. — 978-5-4488-0137-2. — Режим доступа: http://www.iprbookshop.ru/64055.html

4. Малиновская Е.А. Языки программирования. Часть 1 [Электронный ресурс] : лабораторный практикум / Е.А. Малиновская, Р.А. Рыскаленко. — Электрон. текстовые данные. — Ставрополь: Северо-Кавказский федеральный университет, 2016. — 103 c. — 2227-8397. — Режим доступа: http://www.iprbookshop.ru/69449.html

5. Страуструп Б. Язык программирования C++ для профессионалов [Электронный ресурс] / Б. Страуструп. — 2-е изд. — Электрон. текстовые данные. — М. : Интернет-Университет Информационных Технологий (ИНТУИТ), 2016. — 670 c. — 2227-8397. — Режим доступа: http://www.iprbookshop.ru/73737.html

6. Задачник-практикум по основам программирования [Электронный ресурс]: учебное пособие по курсу «Информатика»/ Н.И. Амелина [и др.].— Электрон. текстовые данные.— Ростов-на-Дону: Южный федеральный университет, 2009.— 192 с.— Режим доступа: http://www.iprbookshop.ru/46954.html

VI. Перечень ресурсов информационно-телекоммуникационной сети «Интернет», необходимых для освоения дисциплины (или модуля)

1. ЭБС Лань https://e.lanbook.com/ Договор № 4-е/23 от 02.08.2023 г.
2. ЭБС Znanium.com https://znanium.com/ Договор № 1106 эбс от 02.08.2023 г.
3. ЭБС Университетская библиотека online https://biblioclub.ru Договор № 02-06/2023 от 02.08.2023 г.
4. ЭБС ЮРАЙТ https://urait.ru/ Договор № 5-е/23 от 02.08.2023 г.
5. ЭБС IPR SMART https://www.iprbookshop.ru/ Договор № 3-е/23К от 02.08.2023 г.

VII. Методические указания для обучающихся по освоению дисциплины (или модуля)

Материал дисциплины распределен по главным разделам (темам). В результате изучения дисциплины у студентов должно сформироваться научное представление о системах программирования. Необходимо выработать системный подход к пониманию процессов разработки компьютерных приложений. В процессе обучения студенты, наряду с текстами лекций и учебными пособиями, должны пользоваться дополнительными научными изданиями, академическими периодическими изданиями. После каждой лекционной темы рекомендуется проработать вопросы для повторения и самоконтроля. В аспекте самостоятельной работы рекомендуется составлять конспект с наиболее важными методами и приемами создания приложений. Рекомендуется использовать справочники и руководства.

Для успешного освоения дисциплины важно соблюсти следующие рекомендации: На первой лекции важно обратить внимание на конкретные требования к прохождению и сдаче курса. Активная работа на занятиях, выполнение творческих заданий сформирует о Вас дополнительное положительное представление как об активном участнике познавательного процесса. На данном курсе практические занятия являются самым важным компонентом обучающего процесса. На занятиях будет представлен необходимый теоретически материал по темам и представлены практические задания для решения на занятиях в аудитории под руководством

преподавателя и самостоятельно. Многие задачи являются стандартными и имеют уже готовые шаблоны (алгоритмы) решения, тем не менее, для получения большего познавательного и учебного эффекта, настоятельно рекомендуется написание собственного оригинального кода.

Самостоятельная работа студентов в рамках данной дисциплины в основном состоит в подготовке к практическим занятиям и написании алгоритмов и программ, в работе с разными источниками. Освоению учебного материала большую помощь окажет личный творческий подход, связанный с дополнительным просмотром материала по отдельным темам в библиотеках и системе «Интернет». Самостоятельная работа является необходимой на всей стадиях и при всех формах изучения предмета. Важно помнить: без самостоятельной работы невозможно серьезное освоение любого курса. Надо быть готовым к тому, что по времени, затраченном на дисциплину, самостоятельная работа будет превалировать над иными видами работы. Важно продумать стиль фиксации нового и важного материала. Рекомендуется немедленно обсуждать любые возникшие в процессе обучения вопросы, проблемы и неясности с преподавателем, не откладывая это обсуждение до контрольной точки. Проконсультироваться с преподавателем можно во время и после практических занятий, во время консультаций, а также по электронной почте.

VIII. Перечень педагогических и информационных технологий, используемых при осуществлении образовательного процесса по дисциплине (или модулю), включая перечень программного обеспечения и информационных справочных систем (по необходимости)

Процесс изучения дисциплины включает лекции, лабораторные занятия и самостоятельную работу студента. Во время обучения применяется контактная технология преподавания (за исключением самостоятельно изучаемых студентами вопросов). При проведении занятий применяется имитационный подход (метод деловой игры, анализ конкретных ситуаций), когда преподавателем разбирается на конкретном примере проблемная ситуация, все шаги решения задачи студентам демонстрируются при помощи мультимедийной техники. Затем студенты самостоятельно решают аналогичные задания. Так же при проведении занятий применяется частично-поисковый метод: студенты осуществляют поиск решения поставленной проблемы (задачи). При этом постановочные задачи опираются на уже имеющиеся у студентов знания и умения, полученные в предшествующих темах. На занятиях практикуется выполнение заданий в малых группах, письменные работы, работа с раздаточным материалом, привлекаются ресурсы сети Интернет. Курс предусматривает выполнение тестов, контрольных и самостоятельных работ, письменных домашних заданий. В качестве форм контроля используются различные варианты взаимопроверки и взаимоконтроля.

Программное обеспечение

Adobe Acrobat Reader DC - Russian	бесплатно
Cadence SPB/OrCAD 16.6	Государственный контракт на поставку лицензионных программных продуктов 103 - ГК/09 от 15.06.2009
Google Chrome	бесплатно
Java SE Development Kit 8 Update 45 (64-bit)	бесплатно

Kaspersky Endpoint Security 10 для Windows	Акт на передачу прав ПК545 от 16.12.2022
Lazarus 1.4.0	бесплатно
Mathcad 15 M010	Акт предоставления прав ИС00000027 от 16.09.2011;
MATLAB R2012b	Акт предоставления прав № Us000311 от 25.09.2012;
Mercurial 3.7.3	бесплатно
Microsoft SQL Server 2012 Express LocalDB	бесплатно
Microsoft Web Deploy 3.5	бесплатно
MiKTeX 2.9	бесплатно
MSXML 4.0 SP2 Parser and SDK	бесплатно
MySQL Workbench 6.3 CE	бесплатно
NetBeans IDE 8.0.2	бесплатно
Notepad++	бесплатно
Origin 8.1 Sr2	договор №13918/M41 от 24.09.2009 с ЗАО «СофтЛайн Трейд»;
Python 3.4.3	бесплатно
WinDjView 2.1	бесплатно
WCF RIA Services V1.0 SP2	бесплатно
Многофункциональный редактор ONLYOFFICE бесплатное ПО	бесплатно
ОС Linux Ubuntu бесплатное ПО	бесплатно

IX. Материально-техническая база, необходимая для осуществления образовательного процесса по дисциплине (или модулю)

Учебная лекционная аудитория, оснащенные средствами мультимедиа:

- наличие мультимедиа проектора
- компьютера

Лабораторные занятия должны поводиться в компьютерном классе, оснащенном персональными компьютерами и средствами доступа в Интернет. На занятиях студенты используют аппаратные и программные средства по различным разделам изучаемой дисциплины, а также выполняют лабораторные работы, направленные на закрепление полученных знаний и формирование умений и навыков по их применению.

X. Сведения об обновлении рабочей программы дисциплины (или модуля)

№п.п.	Обновленный раздел рабочей программы дисциплины (или модуля)	Описание внесенных изменений	Дата и протокол заседания кафедры, утвердившего изменения
1.	I - X	14.05.2017 Корректировка всех разделов в соответствии с новым стандартом	
2.			
3.			