

Документ подписан простой электронной подписью
Информация о владельце:
ФИО: Умаров Марат Файзуллаевич
Должность: Директор
Дата подписания: 25.02.2026 14:23:08
Уникальный программный ключ:
48505f11ec15acaa386f5219d3113d727fefda78

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования
"Казанский (Приволжский) федеральный университет"
Елабужский институт (филиал) КФУ



Программа дисциплины (модуля)
Программирование

Направление подготовки/специальность: 44.03.01 Педагогическое образование
Направленность (профиль) подготовки: Технология и робототехника
Квалификация выпускника: бакалавр
Форма обучения: очная
Язык обучения: русский
Год начала обучения по образовательной программе: 2022

Содержание

1. Перечень планируемых результатов обучения по дисциплине (модулю), соотнесенных с планируемыми результатами освоения ОПОП ВО
2. Место дисциплины (модуля) в структуре ОПОП ВО
3. Объем дисциплины (модуля) в зачетных единицах с указанием количества часов, выделенных на контактную работу обучающихся с преподавателем (по видам учебных занятий) и на самостоятельную работу обучающихся
4. Содержание дисциплины (модуля), структурированное по темам (разделам) с указанием отведенного на них количества академических часов и видов учебных занятий
 - 4.1. Структура и тематический план контактной и самостоятельной работы по дисциплине (модулю)
 - 4.2. Содержание дисциплины (модуля)
5. Перечень учебно-методического обеспечения для самостоятельной работы обучающихся по дисциплине (модулю)
6. Фонд оценочных средств по дисциплине (модулю)
7. Перечень литературы, необходимой для освоения дисциплины (модуля)
8. Перечень ресурсов информационно-телекоммуникационной сети "Интернет", необходимых для освоения дисциплины (модуля)
9. Методические указания для обучающихся по освоению дисциплины (модуля)
10. Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине (модулю), включая перечень программного обеспечения и информационных справочных систем (при необходимости)
11. Описание материально-технической базы, необходимой для осуществления образовательного процесса по дисциплине (модулю)
12. Средства адаптации преподавания дисциплины (модуля) к потребностям обучающихся инвалидов и лиц с ограниченными возможностями здоровья
13. Приложение №1. Фонд оценочных средств
14. Приложение №2. Перечень литературы, необходимой для освоения дисциплины (модуля)
15. Приложение №3. Перечень информационных технологий, используемых для освоения дисциплины (модуля), включая перечень программного обеспечения и информационных справочных систем

Программу дисциплины разработал(а)(и) к.н. Анисимова Э.С. (Кафедра математики и прикладной информатики, отделение математики и естественных наук), ESAnisimova@kpfu.ru)

1. Перечень планируемых результатов обучения по дисциплине (модулю), соотнесенных с планируемыми результатами освоения ОПОП ВО

Обучающийся, освоивший дисциплину (модуль), должен обладать следующими компетенциями:

Шифр компетенции	Расшифровка приобретаемой компетенции
ОПК-9	Способен понимать принципы работы современных информационных технологий и использовать их для решения задач профессиональной деятельности
ОПК-9.1	Знать принципы работы современных информационных технологий и способы их использования для решения задач профессиональной деятельности
ОПК-9.2	Уметь применять принципы работы современных информационных технологий и использовать их для решения задач профессиональной деятельности
ОПК-9.3	Владеть пониманием принципов работы современных информационных технологий и навыками их использования для решения задач профессиональной деятельности

Обучающийся, освоивший дисциплину (модуль):

Должен знать:

основные принципы работы современных информационных технологий и способы их использования для решения стандартных задач по программированию.

Должен уметь:

применять основные принципы работы современных информационных технологий и использовать их для решения стандартных задач по программированию.

Должен владеть:

пониманием основных принципов работы современных информационных технологий и навыками их использования для решения стандартных по программированию.

2. Место дисциплины (модуля) в структуре ОПОП ВО

Данная дисциплина (модуль) включена в раздел "Б1.О.08.03 Дисциплины (модули)" основной профессиональной образовательной программы 44.03.01 "Педагогическое образование (Технология и робототехника)" и относится к обязательным дисциплинам. Осваивается на 3 курсе в 4 семестре.

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

Общая трудоемкость дисциплины составляет 5 зачетных(ые) единиц(ы) на 180 часа(ов).

Контактная работа - 90 часа(ов), в том числе лекции - 36 часа(ов), практические занятия - 0 часа(ов), лабораторные работы - 54 часа(ов), контроль самостоятельной работы - 0 часа(ов).

Самостоятельная работа - 54 часа(ов).

Контроль (зачёт / экзамен) - 36 часа(ов).

Форма промежуточного контроля дисциплины: экзамен в 4 семестре.

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

4.1 Структура и тематический план контактной и самостоятельной работы по дисциплине (модулю)

N	Разделы дисциплины / модуля	Семестр	Виды и часы контактной работы, их трудоемкость (в часах)			Самостоятельная работа
			Лекции	Практические занятия	Лабораторные работы	
1.	Тема 1. Основы языка C++	4	4	0	4	4
2.	Тема 2. Управляющие конструкции	4	4	0	4	4
3.	Тема 3. Указатели, ссылки, массивы и текстовые строки	4	4	0	4	4
4.	Тема 4. Функции	4	4	0	3	3
5.	Тема 5. Структуры, объединения и перечисления	4	4	0	3	3
6.	Тема 6. Классы и объекты	4	4	0	10	10
7.	Тема 7. Конструкторы и деструкторы	4	4	0	8	8
8.	Тема 8. Перегрузка операторов	4	4	0	8	8
9.	Тема 9. Наследование и виртуальные функции	4	4	0	10	10
	Итого: 180 часов (из них 36 часов контроль)		36	0	54	54

4.2 Содержание дисциплины (модуля)

Тема 1. Основы языка C++

Структурное программирование в C++. Создание простой программы. Объявление и инициализация переменной. Базовые типы данных. Константы и литералы. Арифметические операторы. Логические операторы. Операторы сравнения. Оператор присваивания и приведение типов. Тернарный оператор. Побитовые операторы и двоичное представление чисел.

Тема 2. Управляющие конструкции

Управляющие инструкции. Ветвления и циклы. Условный оператор IF(). Вложенные условные операторы. Условный оператор SWITCH(). Оператор цикла FOR(). Оператор цикла WHILE(). Синтаксис вызова операторов. Инструкция безусловного перехода. Примеры решения задач. Решение уравнения методом последовательных итераций.

Тема 3. Указатели, ссылки, массивы и текстовые строки

Объявление и использование указателей. Адресная арифметика и сравнение указателей. Многоуровневая адресация. Знакомство со ссылками. Статические одномерные массивы. Указатель на массив. Двумерные массивы. Инициализация массивов. Массивы массивов. Массивы указателей. Создание и инициализация строк. Строчные литералы. Двумерные символьные массивы. Динамические массивы.

Тема 4. Функции

Объявление и использование функций. Механизмы передачи аргументов. Передача указателя аргументом функции. Передача массива аргументом функции. Передача строки аргументом функции. Аргументы функции MAIN(). Аргументы по умолчанию. Возвращение функцией указателя. Возвращение функцией ссылки. Указатели на функции. Рекурсия. Перегрузка функций.

Тема 5. Структуры, объединения и перечисления

Структуры. Массивы структур. Передача структур аргументами функций. Указатели на структуры. Битовые размеры поля. Объединения. Перечисления и определение типов. Примеры решения задач: бинарное дерево, комплексные числа, комплексная экспонента, расстояние между точками, пересечение прямых, корни квадратного уравнения.

Тема 6. Классы и объекты

Объектно-ориентированное программирование. Классы. Объявление класса. Открытые и закрытые члены класса. Статические члены класса. Перегрузка методов. Передача объектов аргументами. Возвращение результатом объектов. Указатели на объекты. Указатели на члены класса. Использование ссылок на объекты. Массивы объектов. Дружественные функции и классы.

Тема 7. Конструкторы и деструкторы

Конструкторы. Создание и перегрузка конструктора. Деструкторы. Правила создания деструктора.

Использование деструкторов. Вызов конструктора. Конструктор создания копии. Примеры решения задач: создание бинарного дерева, ряд для экспоненты, поле-объект, поле-массив объектов, вызов в конструкторе метода.

Тема 8. Перегрузка операторов

Внешняя операторная функция для переопределения бинарного оператора. Перегрузка операторной функции. Переопределение унарных операторов внешними функциями. Перегрузка операторов методами класса. Перегрузка оператора присваивания. Примеры решения задач: индексирование объектов, скалярное и векторное произведение векторов.

Тема 9. Наследование и виртуальные функции

Наследование классов и типы наследования. Переопределение методов и виртуальные функции. Многоуровневое наследование. Конструкторы и деструкторы при наследовании. Чисто виртуальные методы и абстрактные классы. Существующие и несуществующие члены класса. Примеры решения задач: комплексные числа, наследование операторных функций, преобразование Фурье.

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

Самостоятельная работа обучающихся выполняется по заданию и при методическом руководстве преподавателя, но без его непосредственного участия. Самостоятельная работа подразделяется на самостоятельную работу на аудиторных занятиях и на внеаудиторную самостоятельную работу. Самостоятельная работа обучающихся включает как полностью самостоятельное освоение отдельных тем (разделов) дисциплины, так и проработку тем (разделов), осваиваемых во время аудиторной работы. Во время самостоятельной работы обучающиеся читают и конспектируют учебную, научную и справочную литературу, выполняют задания, направленные на закрепление знаний и отработку умений и навыков, готовятся к текущему и промежуточному контролю по дисциплине.

Организация самостоятельной работы обучающихся регламентируется нормативными документами, учебно-методической литературой и электронными образовательными ресурсами, включая:

Порядок организации и осуществления образовательной деятельности по образовательным программам высшего образования – программам бакалавриата, программам специалитета, программам магистратуры (утвержденный приказом Министерства науки и высшего образования Российской Федерации от 6 апреля 2021 года № 245)

Письмо Министерства образования Российской Федерации №14-55-99бин/15 от 27 ноября 2002 г. "Об активизации самостоятельной работы студентов высших учебных заведений"

Устав федерального государственного автономного образовательного учреждения "Казанский (Приволжский) федеральный университет"

Правила внутреннего распорядка федерального государственного автономного образовательного учреждения высшего профессионального образования "Казанский (Приволжский) федеральный университет"

Локальные нормативные акты Казанского (Приволжского) федерального университета

6. Фонд оценочных средств по дисциплине (модулю)

Фонд оценочных средств по дисциплине (модулю) включает оценочные материалы, направленные на проверку освоения компетенций, в том числе знаний, умений и навыков. Фонд оценочных средств включает оценочные средства текущего контроля и оценочные средства промежуточной аттестации.

В фонде оценочных средств содержится следующая информация:

- соответствие компетенций планируемым результатам обучения по дисциплине (модулю);
- критерии оценивания сформированности компетенций;
- механизм формирования оценки по дисциплине (модулю);
- описание порядка применения и процедуры оценивания для каждого оценочного средства;
- критерии оценивания для каждого оценочного средства;
- содержание оценочных средств, включая требования, предъявляемые к действиям обучающихся, демонстрируемым результатам, задания различных типов.

Фонд оценочных средств по дисциплине находится в Приложении 1 к программе дисциплины (модулю).

7. Перечень литературы, необходимой для освоения дисциплины (модуля)

Освоение дисциплины (модуля) предполагает изучение учебной литературы. Литература может быть доступна обучающимся в одном из двух вариантов (либо в обоих из них):

- в электронном виде - через электронные библиотечные системы на основании заключенных КФУ договоров с правообладателями;

- в печатном виде - в Научной библиотеке Елабужского института КФУ. Обучающиеся получают учебную литературу на абонементе по читательским билетам в соответствии с правилами пользования Научной библиотекой.

Электронные издания доступны дистанционно из любой точки при введении обучающимся своего логина и пароля от личного кабинета в системе "Электронный университет". При использовании печатных изданий библиотечный фонд должен быть укомплектован ими из расчета не менее 0,25 экземпляра на каждого обучающегося из числа лиц, одновременно осваивающих данную дисциплину.

Перечень учебной литературы, необходимой для освоения дисциплины (модуля), находится в Приложении 2 к рабочей программе дисциплины. Он подлежит обновлению при изменении условий договоров КФУ с правообладателями электронных изданий и при изменении комплектования фондов Научной библиотеки Елабужского института КФУ.

8. Перечень ресурсов информационно-телекоммуникационной сети "Интернет", необходимых для освоения дисциплины (модуля)

Основы программирования на C++ для начинающих - <https://purecodecpp.com/>

Курс 'Продвинутый C++' - <https://tproger.ru/video/advanced-cpp/>

Уроки программирования на C++ с нуля - <https://code-live.ru/tag/cpp-manual/>

9. Методические указания для обучающихся по освоению дисциплины (модуля)

Вид работ	Методические рекомендации
лекции	В ходе лекционных занятий следует вести конспектирование учебного материала. Обращать внимание на категории, формулировки, раскрывающие содержание темы, научные выводы и практические рекомендации. Желательно оставить в рабочих конспектах поля, на которых делать пометки из рекомендованной литературы, дополняющие материал прослушанной лекции, а также подчеркивающие особую важность тех или иных теоретических положений. Задавать преподавателю уточняющие вопросы с целью уяснения теоретических положений, практических рекомендаций, разрешения проблемных ситуаций. В ходе подготовки к лекционным занятиям повторить изложенный ранее учебный материал, ознакомиться с основной и дополнительной литературой, информацией из рекомендованных Интернет-ресурсов по изученной теме. Дорабатывать свой конспект лекции, делая в нем соответствующие записи из рекомендованной основной и дополнительной литературы, Интернет-ресурсов по проблемным вопросам.
лабораторные работы	Выполнение лабораторных работ направлено на обобщение, систематизацию, углубление теоретических знаний; формирование умений применять полученные знания в практической деятельности; развитие аналитических, проектировочных, конструктивных умений; выработку самостоятельности, ответственности и творческой инициативы. В ходе выполнения лабораторной работы студент должен проявить умение самостоятельно работать с учебной и научной литературой, Интернет-ресурсами, продемонстрировать навыки владения компьютерной техникой и пакетами прикладных программ соответствующего назначения. Контрольной точкой лабораторной работы является ее защита. Защита проводится в устной форме: студент должен уметь объяснить и обосновать каждый выполненный этап работы.
самостоятельная работа	Самостоятельная работа по данной дисциплине включает: повторение теоретического материала; подготовка к практическим занятиям; решение задач, выданных на практических занятиях; подготовка к письменным работам, подготовка к экзаменам. Любая форма самостоятельной работы начинается с изучения конспекта лекции, соответствующей учебной и научной литературы, а также информации из рекомендованных Интернет-ресурсов. Во всех рекомендуемых учебниках и учебных пособиях содержатся контрольные вопросы, которые помогают повторить ключевые моменты соответствующей темы, и практические задания, нацеленные на выявление логических взаимосвязей.
экзамен	Экзамен проводится в устной форме по билетам, в которых содержатся вопросы (задания) по всему разделу дисциплины. Оценивается владение теоретическим материалом, его системное освоение, взаимосвязь основных понятий дисциплины, способность применять знания и умения при решении практических заданий, приобретение навыков самостоятельной работы. Для подготовки к экзамену рекомендуется повторить весь учебный материал по дисциплине, а также использовать основную и дополнительную литературу, информацию из рекомендованных Интернет-ресурсов.

10. Перечень информационных технологий, используемых при осуществлении образовательного

процесса по дисциплине (модулю), включая перечень программного обеспечения и информационных справочных систем (при необходимости)

Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине (модулю), включая перечень программного обеспечения и информационных справочных систем, представлен в Приложении 3 к рабочей программе дисциплины (модуля).

11. Описание материально-технической базы, необходимой для осуществления образовательного процесса по дисциплине (модулю)

Учебная аудитория № 60 (423600, Республика Татарстан, г. Елабуга, ул. Казанская, д. 89) для проведения занятий лекционного типа, занятий семинарского типа, групповых и индивидуальных консультаций, текущего контроля и промежуточной аттестации, помещение для самостоятельной работы. Комплект мебели (посадочных мест) 29 шт. Комплект мебели (посадочных мест) для преподавателя 1 шт. Компьютерный класс: Компьютеры intel core i5 15 шт. Мониторы ViewSonic 22d 15 шт. Проектор EPSON EB-535W 1 шт. Интерактивная доска IQBoard DVT TN082 1 шт. Трибуна 1 шт. Кондиционер 1 шт. Настенные полки 6 шт. Шкаф двухстворчатый с полками 1 шт. Веб-камера 1 шт. Выход в Интернет, внутривизовская компьютерная сеть, доступ в электронную информационно-образовательную среду. Набор учебно-наглядных пособий: комплект презентаций в электронном формате по преподаваемой дисциплине 3-5 шт.

Учебная аудитория № 27 (423600, Республика Татарстан, г. Елабуга, ул. Казанская, д. 89) для проведения занятий семинарского типа, групповых и индивидуальных консультаций, текущего контроля и промежуточной аттестации, помещение для самостоятельной работы. Комплект мебели (посадочных мест) 24 шт. Комплект мебели (посадочных мест) для преподавателя 1 шт. Компьютерный класс: компьютеры Intel(R)Core(TM)i3 9 шт. Мониторы Acer 21,5" 9 шт. Интерактивная доска 1 шт. Парты 8 шт. Шкаф с полочками 1 шт. Меловая доска передвижная 1 шт. Выход в Интернет, внутривизовская компьютерная сеть, доступ в электронную информационно-образовательную среду.

Учебная аудитория №28 (423600, Республика Татарстан, г. Елабуга, ул. Казанская, д. 89) для проведения занятий семинарского типа, групповых и индивидуальных консультаций, текущего контроля и промежуточной аттестации, помещение для самостоятельной работы. Комплект мебели (посадочных мест) 13 шт. Комплект мебели (посадочных мест) для преподавателя 1 шт. Компьютерный класс: компьютеры Intel(R)Core(TM)i5 10 шт. Мониторы Acer 21,5" 10 шт. Парта 1 шт. Шкаф с полками 2 шт. Маркерная доска передвижная 1 шт. Тумба закрытая с двумя дверцами 1 шт. Выход в Интернет, внутривизовская компьютерная сеть, доступ в электронную информационно-образовательную среду.

Учебная аудитория №23 (423600, Республика Татарстан, г. Елабуга, ул. Казанская, д. 89) для проведения занятий семинарского типа, групповых и индивидуальных консультаций, текущего контроля и промежуточной аттестации, помещение для самостоятельной работы. Комплект мебели (посадочных мест) 13 шт. Комплект мебели (посадочных мест) для преподавателя 1 шт. Компьютерный класс: компьютеры Intel(R)Core(TM)i5 10 шт. Мониторы Acer 21,5" 10 шт. Парта 1 шт. Передвижная доска белая магнитная 1 шт. Шкаф закрытый 1 шт. Шкаф с полочками 1 шт. Выход в Интернет, внутривизовская компьютерная сеть, доступ в электронную информационно-образовательную среду.

12. Средства адаптации преподавания дисциплины к потребностям обучающихся инвалидов и лиц с ограниченными возможностями здоровья

При необходимости в образовательном процессе применяются следующие методы и технологии, облегчающие восприятие информации обучающимися инвалидами и лицами с ограниченными возможностями здоровья:

- создание текстовой версии любого нетекстового контента для его возможного преобразования в альтернативные формы, удобные для различных пользователей;
- создание контента, который можно представить в различных видах без потери данных или структуры, предусмотреть возможность масштабирования текста и изображений без потери качества, предусмотреть доступность управления контентом с клавиатуры;
- создание возможностей для обучающихся воспринимать одну и ту же информацию из разных источников - например, так, чтобы лица с нарушениями слуха получали информацию визуально, с нарушениями зрения - аудиально;
- применение программных средств, обеспечивающих возможность освоения навыков и умений, формируемых дисциплиной, за счёт альтернативных способов, в том числе виртуальных лабораторий и симуляционных технологий;
- применение дистанционных образовательных технологий для передачи информации, организации различных форм интерактивной контактной работы обучающегося с преподавателем, в том числе вебинаров, которые могут быть использованы для проведения виртуальных лекций с возможностью взаимодействия всех участников дистанционного обучения, проведения семинаров, выступления с докладами и защиты выполненных работ, проведения тренингов, организации коллективной работы;

- применение дистанционных образовательных технологий для организации форм текущего и промежуточного контроля;

- увеличение продолжительности сдачи обучающимся инвалидом или лицом с ограниченными возможностями здоровья форм промежуточной аттестации по отношению к установленной продолжительности их сдачи:

- продолжительности сдачи зачёта или экзамена, проводимого в письменной форме, - не более чем на 90 минут;

- продолжительности подготовки обучающегося к ответу на зачёте или экзамене, проводимом в устной форме, - не более чем на 20 минут;

- продолжительности выступления обучающегося при защите курсовой работы - не более чем на 15 минут.

Программа составлена в соответствии с требованиями ФГОС ВО и учебным планом по направлению 44.03.01 "Педагогическое образование)" и профилю подготовки "Технология и робототехника".

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования
"Казанский (Приволжский) федеральный университет"
Елабужский институт (филиал)

**Фонд оценочных средств по дисциплине
Б1.О.08.03 Программирование**

Направление подготовки: 44.03.01 - Педагогическое образование

Профиль подготовки: Технология и робототехника

Квалификация выпускника: бакалавр

Форма обучения: очная

Язык обучения: русский

Год начала обучения по образовательной программе: 2022

СОДЕРЖАНИЕ

1. Соответствие компетенций планируемым результатам обучения по дисциплине (модулю)
2. Критерии оценивания сформированности компетенций
3. Распределение оценок за формы текущего контроля и промежуточную аттестацию
4. Оценочные средства, порядок их применения и критерии оценивания
 - 4.1. Оценочные средства текущего контроля
 - 4.1.1. Тестирование
 - 4.1.1.1. Порядок проведения.
 - 4.1.1.2. Критерии оценивания
 - 4.1.1.3. Содержание оценочного средства
 - 4.1.2. Лабораторные работы
 - 4.1.2.1. Порядок проведения.
 - 4.1.2.2 Критерии оценивания
 - 4.1.2.3. Содержание оценочного средства
 - 4.1.3. Контрольная работа
 - 4.1.3.1. Порядок проведения.
 - 4.1.3.2 Критерии оценивания
 - 4.1.3.3. Содержание оценочного средства
 - 4.2. Оценочные средства промежуточной аттестации
Экзамен
 - 4.2.1. Устный или письменный ответ на вопрос
 - 4.2.1.1. Порядок проведения и процедура оценивания.
 - 4.2.1.2. Критерии оценивания.
 - 4.2.1.3. Оценочные средства.
 - 4.2.2. Практическое задание
 - 4.2.2.1. Порядок проведения и процедура оценивания
 - 4.2.2.2. Критерии оценивания.
 - 4.2.2.3. Оценочные средства.

1. Соответствие компетенций планируемым результатам обучения по дисциплине (модулю)

Код и наименование компетенции	Проверяемые результаты обучения для данной дисциплины	Оценочные средства текущего контроля и промежуточной аттестации
<p>ОПК-9 Способен понимать принципы работы современных информационных технологий и использовать их для решения задач профессиональной деятельности</p>	<p>Знать основные принципы работы современных информационных технологий и способы их использования для решения стандартных задач по программированию.</p> <p>Уметь применять основные принципы работы современных информационных технологий и использовать их для решения стандартных задач по программированию.</p> <p>Владеть пониманием основных принципов работы современных информационных технологий и навыками их использования для решения стандартных по программированию.</p>	<p>Текущий контроль: Тестирование по темам Тема 1. Основы языка C++ Тема 2. Управляющие конструкции Тема 3. Указатели, ссылки, массивы и текстовые строки Тема 4. Функции Тема 5. Структуры, объединения и перечисления Тема 6. Классы и объекты Тема 7. Конструкторы и деструкторы Тема 8. Перегрузка операторов Тема 9. Наследование и виртуальные функции</p> <p>Лабораторные работы по темам Тема 1. Основы языка C++ Тема 2. Управляющие конструкции Тема 3. Указатели, ссылки, массивы и текстовые строки Тема 4. Функции Тема 5. Структуры, объединения и перечисления Тема 6. Классы и объекты Тема 7. Конструкторы и деструкторы Тема 8. Перегрузка операторов Тема 9. Наследование и виртуальные функции</p> <p>Контрольная работа по темам Тема 1. Основы языка C++ Тема 2. Управляющие конструкции Тема 3. Указатели, ссылки, массивы и текстовые строки Тема 4. Функции Тема 5. Структуры, объединения и перечисления Тема 6. Классы и объекты Тема 7. Конструкторы и деструкторы Тема 8. Перегрузка операторов Тема 9. Наследование и виртуальные функции</p> <p>Промежуточная аттестация: <i>Экзамен</i></p>

2. Критерии оценивания сформированности компетенций

Компетенция	Зачтено			Не зачтено
	Высокий уровень (отлично) (86-100 баллов)	Средний уровень (хорошо) (71-85 баллов)	Низкий уровень (удовлетворительно) (56-70 баллов)	Ниже порогового уровня (неудовлетворительно) (0-55 баллов)
ОПК-9	Знает основные принципы работы современных информационных технологий и способы их использования для решения	Знает основные принципы работы современных информационных технологий и способы их	Знает отдельные принципы работы современных информационных технологий и способы	Не знает основные принципы работы современных информационных технологий и способы

Компетенция	Зачтено			Не зачтено
	Высокий уровень (отлично) (86-100 баллов)	Средний уровень (хорошо) (71-85 баллов)	Низкий уровень (удовлетворительно) (56-70 баллов)	Ниже порогового уровня (неудовлетворительно) (0-55 баллов)
	стандартных задач по программированию	использования для решения стандартных задач по программированию, допускает незначительные ошибки при ответе на вопрос или решении поставленной задачи	их использования для решения стандартных задач по программированию, допускает типичные ошибки при ответе на вопрос или решении поставленной задачи	их использования для решения стандартных задач по программированию
	Умеет применять основные принципы работы современных информационных технологий и использовать их для решения стандартных задач по программированию	Умеет применять основные принципы работы современных информационных технологий и использовать их для решения стандартных задач по программированию, допускает незначительные ошибки при ответе на вопрос или решении поставленной задачи	Умеет применять основные принципы работы современных информационных технологий и использовать их для решения стандартных задач по программированию, допускает типичные ошибки при ответе на вопрос или решении поставленной задачи	Не умеет применять основные принципы работы современных информационных технологий и использовать их для решения стандартных задач по программированию
	Владеет пониманием основных принципов работы современных информационных технологий и навыками их использования для решения стандартных задач по программированию	Владеет пониманием основных принципов работы современных информационных технологий и навыками их использования для решения стандартных задач по программированию, допускает незначительные ошибки при ответе на вопрос или решении поставленной задачи	Владеет пониманием основных принципов работы современных информационных технологий и навыками их использования для решения стандартных задач по программированию, допускает типичные ошибки при ответе на вопрос или решении поставленной задачи	Не владеет пониманием основных принципов работы современных информационных технологий и навыками их использования для решения стандартных задач по программированию

3. Распределение оценок за формы текущего контроля и промежуточную аттестацию

4 семестр:

Текущий контроль:

Текущий контроль:

Тестирование

Тема 1. Основы языка C++

Тема 2. Управляющие конструкции

Тема 3. Указатели, ссылки, массивы и текстовые строки

Тема 4. Функции

Тема 5. Структуры, объединения и перечисления

Тема 6. Классы и объекты

Тема 7. Конструкторы и деструкторы

Тема 8. Перегрузка операторов

Тема 9. Наследование и виртуальные функции

Максимальное количество баллов по БРС - 10.

Лабораторные работы

Тема 1. Основы языка C++

Тема 2. Управляющие конструкции

Тема 3. Указатели, ссылки, массивы и текстовые строки

Тема 4. Функции
Тема 5. Структуры, объединения и перечисления
Тема 6. Классы и объекты
Тема 7. Конструкторы и деструкторы
Тема 8. Перегрузка операторов
Тема 9. Наследование и виртуальные функции

Максимальное количество баллов по БРС - 30.

Контрольная работа

Тема 1. Основы языка C++
Тема 2. Управляющие конструкции
Тема 3. Указатели, ссылки, массивы и текстовые строки
Тема 4. Функции
Тема 5. Структуры, объединения и перечисления
Тема 6. Классы и объекты
Тема 7. Конструкторы и деструкторы
Тема 8. Перегрузка операторов
Тема 9. Наследование и виртуальные функции

Максимальное количество баллов по БРС - 10.

Итого $10+30+10=50$ баллов

Промежуточная аттестация – экзамен – 50 баллов.

Промежуточная аттестация проводится после завершения изучения дисциплины или ее части в форме, определяемой учебным планом образовательной программы с целью оценить работу обучающегося, степень усвоения теоретических знаний, уровень сформированности компетенций.

Преподаватель, принимающий экзамен обеспечивает случайное распределение вариантов зачетных заданий между обучающимися с помощью билетов и/или с применением компьютерных технологий; вправе задавать обучающемуся дополнительные вопросы и давать дополнительные задания помимо тех, которые указаны в билете. Экзамен проводится по билетам. В каждом билете два оценочных средства: устный или письменный ответ на вопрос и практическое задание.

Устный или письменный ответ – 20 баллов.

Практическое задание – 30 баллов.

Итого $20+30=50$ баллов.

Общее количество баллов по дисциплине за текущий контроль и промежуточную аттестацию: $50+50=100$ баллов.

Соответствие баллов и оценок:

Для экзамена

86-100 – отлично.

71-85 – хорошо.

56-70 – удовлетворительно.

0-55 – неудовлетворительно.

4. Оценочные средства, порядок их применения и критерии оценивания

4.1. Оценочные средства текущего контроля

4.1.1. Тестирование

Тема 1. Основы языка C++
Тема 2. Управляющие конструкции
Тема 3. Указатели, ссылки, массивы и текстовые строки
Тема 4. Функции
Тема 5. Структуры, объединения и перечисления
Тема 6. Классы и объекты
Тема 7. Конструкторы и деструкторы
Тема 8. Перегрузка операторов
Тема 9. Наследование и виртуальные функции.

4.1.1.1 Порядок проведения и процедура оценивания

Тестирование проходит в письменной форме или с использованием компьютерных средств. Обучающийся получает определенное количество тестовых заданий. На выполнение выделяется фиксированное время в

зависимости от количества заданий. Оценка выставляется в зависимости от процента правильно выполненных заданий. Тестирование проводится по вариантам. В каждом варианте – 20 тестовых заданий. Ниже приведены примерные задания. Полный банк тестовых заданий хранится на кафедре.

4.1.1.2. Критерии оценивания

9-10 баллов ставится, если обучающийся набрал:

86% правильных ответов и более.

7-8 баллов ставится, если обучающийся набрал:

От 71% до 85 % правильных ответов.

5-6 баллов ставится, если обучающийся набрал:

От 56% до 70% правильных ответов.

0-4 балла ставится, если обучающийся набрал:

55% правильных ответов и менее.

4.1.1.3. Содержание оценочного средства

Тема 1. Основы языка C++

Тема 2. Управляющие конструкции

Тема 3. Указатели, ссылки, массивы и текстовые строки

Тема 4. Функции

Тема 5. Структуры, объединения и перечисления

1. Функция вычисляет произведение двух чисел. Исходные данные вводятся с клавиатуры. Какие проверки целесообразно ввести в программе:

- а) проверка, что исходные данные являются числами
- б) проверки не нужны, все возможные ошибки отловит компилятор
- в) проверка исходных данных на равенство нулю

2. Для чего предназначен оператор namespace:

- а) для использования классов, переменных и функций из других модулей программы без использования заголовочных файлов
- б) для заключения в группу объявлений классов, переменных и функций в отдельный контекст со своим именем
- в) для заключения в группу объявлений классов, переменных и функций для использования только в текущем модуле

3. Какой из компонентов может входить в интегрированную среду программирования:

- а) наладчик
- б) доводчик
- в) отладчик

4. Какой из компонентов может входить в интегрированную среду программирования:

- а) текстовый редактор
- б) текстовый директор
- в) текстовый модератор

5. Какой из компонентов может входить в интегрированную среду программирования:

- а) регулятор
- б) доминатор
- в) компилятор

6. Если определена операция вычитания для двух объектов класса A, а операция преобразования к int не определена, что будет вызвано при:

A a1,a2,a3=5;

a3 = a1 - a2;

- а) только операция вычитания
- б) произойдет ошибка
- в) преобразование к целому

7. Какой из наборов перечисляемых значений записан правильно:

- а) enum { a, b = 3, c = 4, 3 };
- б) enum { a, b, 3, 4 };
- в) enum {a, b = 3, c, d };

8. В чем различие использования следующих выражений #include <...> и #include «...»:

- а) различие заключается в методе поиска препроцессором включаемого файла
- б) в различии использования заголовочных и исходных файлов
- в) нет различий

9. Чему будет равен результат вычисления выражения: int d=5; bool b = true; c; c = (!b||(d>3)):

- а) Ошибка компилятора
- б) false
- в) true

10. Если в арифметическом выражении участвуют целый и вещественный операнды, то:
- а) ошибка компиляции
 - б) целый тип приводится к вещественному
 - в) вещественный тип приводится к целому
11. Укажите в каком выражении произойдет потеря точности:
- а) `int i; float x = 2.134, y = 3.14; i = x/y;`
 - б) `short i = 0x3; float x = 2.7, v; v = i + x;`
 - в) `float M = 235.2; double Z = 3; Z *= M;`
12. Если после выражения стоит точка с запятой, то:
- а) выражение вычисляется, а его значение запоминается в специальной переменной, которую можно использовать в следующем операторе
 - б) это оператор-выражение, действие которого заключается в вычислении выражения
 - в) выражение вычисляется только если первой стоит операция присваивания
13. Что из себя представляет динамическое выделение памяти:
- а) память под объект (переменную) может выделяться не сразу, а в процессе работы программы, освобождение памяти производится вручную
 - б) память под объект (переменную) может выделяться не сразу, а в процессе работы программы, освобождение памяти производится автоматически после завершения программы
 - в) память под объект (переменную) выделяется каждый раз при обращении к переменной
14. Отметьте истинное высказывание:
- а) переменная инициализируется, потом объявляется
 - б) переменная объявляется, потом инициализируется и изменяется
 - в) переменная объявляется, потом изменяется
15. Какие операции поддаются перегрузке:
- а) унарные и бинарные
 - б) только бинарные
 - в) только унарные
16. Переменная типа `signed char` может принимать значения:
- а) только символов английского алфавита, цифр и символа подчеркивания
 - б) из первой половины кодовой таблицы
 - в) только из алфавита языка C++
17. Переменная типа `signed char` может принимать значения:
- а) только из алфавита языка C++
 - б) только символов английского алфавита, цифр и символа подчеркивания
 - в) от -128 до 127
18. В переменной типа `unsigned char` можно хранить число:
- а) -213
 - б) 213
 - в) 1213
19. В переменной типа `unsigned char` можно хранить число:
- а) -13
 - б) 1213
 - в) 13
20. Чему равно числовое значение выражения $e/2 * a - \text{abs}(e) * 1e0$ при $e = 4, a = 2$:
- а) 3
 - б) 0
 - в) 1
21. Выберите правильное утверждение:
- а) целой переменной можно присвоить вещественную константу
 - б) целой константе можно присвоить целую переменную
 - в) целой константе можно присвоить вещественную переменную
22. Выберите правильное утверждение:
- а) целой константе можно присвоить целую переменную
 - б) целой переменной можно присвоить целую константу
 - в) целой константе можно присвоить вещественную переменную
23. Чему равно значение выражения $(a \ \&\& \ ! \ b \ || \ c)$, где a, b и c - величины типа `bool`, имеющие значения `false, true` и `true` соответственно:
- а) `false`
 - б) `yes`
 - в) `true`
24. Какое выражение не содержит синтаксических ошибок:
- а) `sin(abs(0.6(e*3)))`

б) $((\cos(3*a+1.*\text{abs}(x))))$

в) $a*\exp(t)\backslash(2t)$

25. Какое выражение не содержит синтаксических ошибок:

а) $a*\exp(t)\backslash(2t)$

б) $\sin(\text{abs}(0.6(e*3)))$

в) $0XCC00*.34E-4/_do/k-2$

26. Чему равно числовое значение выражения $\text{sqrt}(4)+142/20*2$:

а) 5

б) 16

в) 9

27. Какая из следующих операций языка C выполняется справа налево:

а) =

б) ->

в) []

28. Какое выражение не содержит синтаксических ошибок:

а) $(-0.18)*a)/(r-0.2*t)$

б) $(-0.18)*a)\backslash(r-0.2*t)$

в) $-0.18*a/r-0.2*t$

29. При использовании в программе функции scanf требуется разделять числовые значения величин при вводе при помощи:

а) нажатия клавиш TAB или ENTER

б) клавиш курсора

в) пробела или запятой

30. При использовании в программе функции scanf требуется разделять числовые значения величин при вводе при помощи:

а) пробела или запятой

б) по крайней мере одного пробела

в) клавиш курсор

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
а	б	в	а	в	б	в	а	в	б	а	б	а	в	в
16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
б	в	б	в	б	а	б	в	б	в	б	а	в	а	б

Тема 6. Классы и объекты

Тема 7. Конструкторы и деструкторы

Тема 8. Перегрузка операторов

Тема 9. Наследование и виртуальные функции

1. Класс - это:

- а) любой тип данных, определяемый пользователем
- б) тип данных, определяемый пользователем и сочетающий в себе данные и функции их обработки
- в) структура, для которой в программе имеются функции работы с нею

2. Выберите правильные утверждения:

- а) по умолчанию члены класса имеют атрибут private
- б) по умолчанию члены класса имеют атрибут public;
- в) члены класса имеют доступ только к элементам public;
- г) элементы класса с атрибутом private доступны только членам класса

3. Переопределение операций имеет вид:

- а) имя_класса, ключевое слово operation, символ операции
- б) имя_класса, ключевое слово operator, символ операции, в круглых скобках могут быть указаны аргументы
- в) имя_класса, ключевое слово operator, список аргументов
- г) имя_класса, два двоеточия, ключевое слово operator, символ операции

4. Объект - это

- а) переменная, содержащая указатель на класс
- б) экземпляр класса
- в) класс, который содержит в себе данные и методы их обработки

5. Тест. Членами класса могут быть

- а) как переменные, так и функции, могут быть объявлены как private и как public
- б) только функции, объявленные как private
- в) только переменные и функции, объявленные как private
- г) только переменные и функции, объявленные как public

6. Что называется конструктором?

- а) метод, имя которого совпадает с именем класса и который вызывается автоматически при создании объекта класса
- б) метод, имя которого совпадает с именем класса и который вызывается автоматически при объявлении класса (до создания объекта класса)
- в) метод, имя которого необязательно совпадает с именем класса и который вызывается при создании объекта класса
- г) метод, имя которого совпадает с именем класса и который необходимо явно вызывать из головной программы при объявлении объекта класса

7. Выберите правильные утверждения

- а) у конструктора могут быть параметры
- б) конструктор наследуется, но должен быть перегружен
- в) конструктор должен явно вызываться всегда перед объявлением объекта
- г) конструктор вызывается автоматически при объявлении объекта
- д) объявление каждого класса должно содержать свой конструктор
- е) если конструктор не создан, компилятор создаст его автоматически

8. Отметьте правильные утверждения

- а) конструкторы класса не наследуются
- б) конструкторов класса может быть несколько, их синтаксис определяется программистом
- в) конструкторов класса может быть несколько, но их синтаксис должен подчиняться правилам перегрузки функций
- г) конструктор возвращает указатель на объект
- д) конструктор не возвращает значение

9. Что называется деструктором?

- а) метод, который уничтожает объект
- б) метод, который удаляет объект
- в) метод, который освобождает память, занимаемую объектом
- г) системная функция, которая освобождает память, занимаемую объектом

10. Выберите правильные утверждения

- а) деструктор - это метод класса, применяемый для удаления объекта
- б) деструктор - это метод класса, применяемый для освобождения памяти, занимаемой объектом
- в) деструктор - это отдельная функция головной программы, применяемая для освобождения памяти, занимаемой объектом
- г) деструктор не наследуется
- д) деструктор наследуется, но должен быть перегружен

11. Что называется наследованием?

- а) это механизм, посредством которого производный класс получает элементы родительского и может дополнять либо изменять их свойства и методы
- б) это механизм переопределения методов базового класса
- в) это механизм, посредством которого производный класс получает все поля базового класса
- г) это механизм, посредством которого производный класс получает элементы родительского, может их дополнить, но не может переопределить

12. Выберите правильное объявление производного класса

- а) `class MoreDetails:: Details;`
- б) `class MoreDetails: public class Details;`
- в) `class MoreDetails: public Details;`
- г) `class MoreDetails: class(Details);`

13. Выберите правильные утверждения:

- а) если элементы класса объявлены как `private`, то они доступны только наследникам класса, но не внешним функциям
- б) если элементы класса объявлены как `private`, то они недоступны ни наследникам класса, ни внешним функциям
- в) если элементы объявлены как `public`, то они доступны наследникам класса, но не внешним функциям
- г) если элементы объявлены как `public`, то они доступны и наследникам класса, и внешним функциям

14. Возможность и способ обращения производного класса к элементам базового определяется

- а) ключами доступа: `private`, `public`, `protected` в теле производного класса
- б) только ключом доступа `protected` в заголовке объявления производного класса
- в) ключами доступа: `private`, `public`, `protected` в заголовке объявления производного класса
- г) ключами доступа: `private`, `public`, `protected` в теле базового класса

15. Выберите правильные соответствия между спецификатором базового класса, ключом доступа в объявлении производного класса и правами доступа производного класса к элементам базового

- а) ключ доступа - `public`; в базовом классе: `private`; права доступа в производном классе - `protected`
- б) ключ доступа - любой; в базовом классе: `private`; права доступа в производном классе - нет прав
- в) ключ доступа - `protected` или `public`; в базовом классе: `protected`; права доступа в производном классе -

protected

г) ключ доступа - private; в базовом классе: public; права доступа в производном классе - public

д) ключ доступа – любой; в базовом классе: public; права доступа в производном классе – такие же, как ключ доступа

16. Дружественная функция - это

а) функция другого класса, среди аргументов которой есть элементы данного класса

б) функция, объявленная в классе с атрибутом friend, но не являющаяся членом класса;

в) функция, являющаяся членом класса и объявленная с атрибутом friend;

г) функция, которая в другом классе объявлена как дружественная данному

17. Выберите правильные утверждения:

а) одна функция может быть дружественной нескольким классам

б) дружественная функция не может быть обычной функцией, а только методом другого класса

в) дружественная функция объявляется внутри класса, к элементам которого ей нужен доступ

г) дружественная функция не может быть методом другого класса

18. Шаблон функции - это...

а) определение функции, в которой типу обрабатываемых данных присвоено условное обозначение

б) прототип функции, в котором вместо имен параметров указан условный тип

в) определение функции, в котором указаны возможные варианты типов обрабатываемых параметров

г) определение функции, в котором в прототипе указан условный тип, а в определении указаны варианты типов обрабатываемых параметров

19. Для доступа к элементам объекта используются:

а) при обращении через имя объекта – точка, при обращении через указатель – операция «->»

б) при обращении через имя объекта – два двоеточия, при обращении через указатель – операция «точка»

в) при обращении через имя объекта – точка, при обращении через указатель – два двоеточия

г) при обращении через имя объекта – два двоеточия, при обращении через указатель – операция «->»

20. Полиморфизм – это :

а) средство, позволяющее использовать одно имя для обозначения действий, общих для родственных классов

б) средство, позволяющее в одном классе использовать методы с одинаковыми именами;

в) средство, позволяющее в одном классе использовать методы с разными именами для выполнения одинаковых действий

г) средство, позволяющее перегружать функции для работы с разными типами или разным количеством аргументов.

21. Полиморфизм реализован через механизмы:

а) перегрузки функций, виртуальных функций, шаблонов

б) перегрузки функций, наследования методов, шаблонов;

в) наследования методов, виртуальных функций, шаблонов

г) перегрузки функций, наследования, виртуальных функций.

22. Виртуальными называются функции:

а) функции базового класса, которые могут быть переопределены в производном классе

б) функции базового класса, которые не используются в производном классе;

в) функции базового класса, которые не могут быть переопределены в базовом классе;

г) функции производного класса, переопределенные относительно базового класса

23. Полиморфизм в объектно-ориентированном программировании реализуется:

а) через механизмы перегрузки (функций и операций), виртуальные функции и шаблоны

б) через механизмы перегрузки (функций и операций) и шаблоны;

в) через виртуальные функции и шаблоны;

г) через механизмы перегрузки (функций и операций) и виртуальные функции

24. Укажите свойства и методы, доступные внешним функциям

а) health, armo

monstr(int he, int arm);

monstr(int he=50, int arm=10);

б) int color;

monstr(int he=50, int arm=10);

в) health, armo, color

monstr(int he=50, int arm=10);

г) int color;

monstr(int he, int arm);

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
б	а,г	б	б	а	а	а,г,е	а,в,д	в	б,г	а	в	б,г	в	б,в,д	б	а,в	а	а	а	а	а	а	б

4.1.2. Лабораторные работы

Тема 1. Основы языка C++

Тема 2. Управляющие конструкции

Тема 3. Указатели, ссылки, массивы и текстовые строки

Тема 4. Функции

Тема 5. Структуры, объединения и перечисления

Тема 6. Классы и объекты

Тема 7. Конструкторы и деструкторы

Тема 8. Перегрузка операторов

Тема 9. Наследование и виртуальные функции

4.1.2.1. Порядок проведения.

Лабораторные работы проводятся в часы аудиторной работы.

Перед выполнением каждой работы студенты-бакалавры должны проработать соответствующий материал, используя конспекты теоретических занятий, периодические издания, учебно-методические пособия и учебники.

По окончании занятий студенты оформляют отчет по каждой работе, соблюдая следующую форму:

- Наименование темы;
- Цель работы;
- Задание и содержание выполненной работы,
- Письменные ответы на контрольные вопросы.
- Выводы по проделанной работе.
- Список использованных источников.

4.1.2.2 Критерии оценивания

26-30 баллов ставится, если обучающийся:

Правильно выполнил все задания. Продемонстрировал высокий уровень владения материалом. Проявлены превосходные способности применять знания и умения к выполнению конкретных заданий.

21-25 баллов ставится, если обучающийся:

Правильно выполнил большую часть заданий. Присутствуют незначительные ошибки. Продемонстрирован хороший уровень владения материалом. Проявлены средние способности применять знания и умения к выполнению конкретных заданий.

17-20 баллов ставится, если обучающийся:

Задания выполнил более чем наполовину. Присутствуют серьезные ошибки. Продемонстрирован удовлетворительный уровень владения материалом. Проявлены низкие способности применять знания и умения к выполнению конкретных заданий.

0-16 баллов ставится, если обучающийся:

Задания выполнил менее чем наполовину. Продемонстрирован неудовлетворительный уровень владения материалом. Проявлены недостаточные способности применять знания и умения к выполнению конкретных заданий.

4.1.2.3. Содержание оценочного средства

4 семестр:

Тема 1. Основы языка C++

Тема 2. Управляющие конструкции

Тема 3. Указатели, ссылки, массивы и текстовые строки

Тема 4. Функции

Тема 5. Структуры, объединения и перечисления

Лабораторная работа 1. Основы языка C++.

1. Введите обозначения и объявите переменные x , y , z и s . Запишите инструкции, которые присваивают переменной "x" значение -25.6, переменной "y" значение 128, а значение переменной "z" вводится с клавиатуры. Вычислите сумму значений трех величин по формуле: $s=x+y+z$.

Выведите результат на экран в виде:

$x=-25.6$

$y=128$

You enter $z=1$

Summa: $s=103.4$

2. Определить тип заданных выражений и найти их значения. Составить систему тестов и вычислить полученное выражение для нескольких значений n и m , определить при каких n и m выражение не может быть вычислено. При выполнении заданий на сравнения, использовать тернарную операцию.

a) $k=n(++m)$;

б) $m \rightarrow n$;

3. Записать выражение, зависящее от координат точки X и Y и принимающее значение TRUE (t или 1), если точка принадлежит выделенной области, и FALSE (f или 0), если не принадлежит. Результаты вычислений вывести на печать. При выполнении задания использовать переменную логического типа, а не условный оператор.

Лабораторная работа 2. Управляющие конструкции.

Часть 1. Написать программу, реализующую алгоритм (в соответствии с вариантом) из лабораторной работы № 1.

Часть 2. Выполнить задания:

1. Объявите переменную $radius$ типа $float$, переменную f типа $float$ со значением 3,14, переменную s – площадь круга, переменную c – длину окружности. Значение $radius$ вводит пользователь.

Рассчитать площадь круга и длину окружности, записать в переменные s и c , вывести результат на экран.

2. Пользователь вводит коэффициенты квадратного уравнения: a , b и

c . Рассчитать дискриминант d . Если дискриминант меньше нуля, вывести сообщение о том, что корней нет. Если дискриминант равен нулю, рассчитать и вывести один корень x . Иначе рассчитать корни x_1 , x_2 .

3. Подсчитать в цикле квадраты и кубы чисел от -5 до 5, вывести на экран.

4. Пользователь вводит числа с клавиатуры до тех пор, пока не будет введено первое отрицательное число.

Вывести на экран сумму чисел, введенных пользователем: а) без учета отрицательного числа; б) с учетом отрицательного числа.

5. Пользователь вводит два числа: x и y и номер действия: 1 – подсчитать сумму, 2 – подсчитать разность. Если введен некорректный номер действия, выводить предупреждающее сообщение. Иначе рассчитать результат и вывести на экран.

Часть 3

Реализовать на C++ калькулятор с возможностью выполнения четырех арифметических действий. У пользователя запрашивать аргументы и знак операции (символ). Программу «заикнуть» – после вывода результата предлагать пользователю произвести вычисления еще раз. Выход из программы производить по нажатию клавиши q .

Осуществить проверки:

1) деления на ноль (в этом случае выводить предупреждающее сообщение и запрашивать делитель еще раз); 2) корректности ввода знака операции (в этом случае просить ввести корректный знак операции).

Пример работы с программой:

Введите первый операнд: 5

Введите второй операнд: 0

Введите знак операции: /

На ноль делить нельзя!

Введите второй операнд: 2

Результат: 2,5

Нажмите любую клавишу для продолжения (выход - q)...

Введите первый операнд: 3

Введите второй операнд: 2

Введите знак операции: %

Некорректно введен знак операции!

Введите знак операции: ?

Некорректно введен знак операции!

Введите знак операции: +

Результат: 5

Нажмите любую клавишу для продолжения (выход - q)...

Лабораторная работа 3. Указатели, ссылки, массивы и текстовые строки.

Указатели и массивы

В языке C/C++ имя (идентификатор) массива является указателем на его нулевой элемент. Другими словами, если, например, описан массив: `int a[10];`, то его имя a – это тоже самое, что и `&a[0]`, а если применить операцию разадресации (*) к имени массива, то получим его нулевой элемент, то есть следующие два выражения эквивалентны: `*a` и `a[0]` также, как и: `a` и `&a[0]`.

Так как элементы массива всегда располагаются в смежных областях памяти, поэтому доступ к каждому i -му элементу можно осуществить, смещая указатель a на i позиций. Другими словами, запись `a[i]` можно заменять на `*(a+i)`. Аналогично можно описать указатель, присвоить ему адрес начала массива и работать с массивом через указатель, смещая указатель на 1, тем самым переходя к очередному элементу массива.

Этот способ проиллюстрирован в следующем примере.

Но существуют и отличия массива от указателя: имя массива не может ни на что указывать; указатель, в отличие от массива, можно перенаправить на любую другую переменную такого же типа, а массив – нет, массив всегда указывает на свой нулевой элемент.

Задание

1. Описать 2 указателя на целый тип. Выделить для них динамическую память. Ввести значения в выделенную память с клавиатуры. Уменьшить в 2 раза 1-ую переменную.
2. Описать 2 указателя на вещественный тип. Выделить для них динамическую память. Ввести значения в выделенную память с клавиатуры. Увеличить в 2 раза 1-ую переменную.
3. Описать 3 указателя на символьный тип. Выделить для них динамическую память. Ввести значения в выделенную память с клавиатуры.
4. Описать 2 указателя на логический тип. Выделить для них динамическую память. Присвоить значения true и false в выделенную память.
5. Описать 2 указателя на целый тип. Выделить для них динамическую память. Присвоить произвольные значения в выделенные ячейки в операторе присвоения.
6. Описать 3 указателя на вещественный тип. Выделить для них динамическую память. Присвоить произвольные значения в выделенные ячейки в операторе присвоения. Уменьшить в 2 раза 1-ую переменную.
7. Описать 1 указатель на символьный тип. Выделить для него динамическую память. Присвоить произвольное значение в выделенную ячейку в операторе присвоения.
8. Описать 2 указателя на целый тип. Выделить для них динамическую память. Ввести значения в выделенную память с клавиатуры. Поменять местами их значения.
9. Описать 2 указателя на вещественный тип. Выделить для них динамическую память. Ввести значения в выделенную память с клавиатуры. Поменять местами их значения.
10. Описать 3 указателя на символьный тип. Выделить для них динамическую память. Ввести значения в выделенную память с клавиатуры. Поменять местами значения первых 2-х переменных.
11. Создать динамические массивы, используя указатели. Задан одномерный массив $a(n)$. Найти количество, все номера и произведение элементов массива меньших
12. Создать динамические массивы, используя указатели. Дано 2 массива $x(n)$ и $y(m)$. Сколько раз встречается второй элемент первого массива $x(n)$ во втором массиве $y(m)$.
13. Создать динамические массивы, используя указатели. В каком из двух данных массивов $p(n)$ $q(n)$ больше отрицательных элементов?
14. Создать динамические массивы, используя указатели. Дан массив $p(n)$. Каждый положительный элемент в нем возвести в квадрат. Остальные элементы оставить прежними.
15. Создать динамические массивы, используя указатели. Задан одномерный массив $a(n)$. Найти номер последнего элемента меньшего заданного числа $beta$, количество положительных элементов и сумму элементов больших 3.
16. Создать динамические массивы, используя указатели. В каком из двух данных массивов $p(n)$ $q(n)$ больше положительных элементов?
17. Создать динамические массивы, используя указатели. Дано 2 массива $x(n)$ и $y(m)$. Сколько раз встречается первый элемент первого массива $x(n)$ во втором массиве $y(m)$.
18. Создать динамические массивы, используя указатели. Дан массив $r(n)$. Каждый элемент равный 0 в нем заменить на 1. Остальные оставить прежними.
19. Создать динамические массивы, используя указатели. Задан одномерный массив $a(n)$. Найти номер последнего элемента равного 5 и переставить его с первым элементом массива. Найти среднее арифметическое элементов массива больших заданного числа $alpha$.
20. Создать динамические массивы, используя указатели. Задан одномерный массив $a(n)$. Найти номер последнего положительного элемента и переставить его с первым элементом массива. Найти количество и сумму элементов отрицательных массива.

Лабораторная работа 4. Функции.

1. Даны числа a, b, c, d . Получить $x = \max(a, b)$, $y = \max(c, d)$, $z = \max(x, y)$. Вычисление $\max(k, m)$ (большого из двух чисел k, m) оформить функцией.
2. Оформить функцию $\text{step}(x, n)$ от вещественного x и целого n , вычисляющую (через последовательное умножение) x^n и проверить ее.
3. Даны координаты вершин двух треугольников. Определить, какой из них имеет большую площадь. Вычисление площади треугольника по координатам оформить функцией.
4. Даны четыре натуральных числа. Найти наименьшее общее кратное (НОК) для этих четырех чисел. Поиск НОК двух чисел оформить функцией.
5. Даны координаты n точек на плоскости в виде массивов X, Y . Найти наиболее и наименее удаленные точки. Вычисление расстояния между парой точек оформить функцией.
6. Даны отрезки a, b, c, d . Для каждой тройки этих отрезков, из которых можно построить треугольник, напечатать площадь данного треугольника. Вычисление площади треугольника по заданным сторонам x, y, z с проверкой существования такого треугольника оформить функцией.
7. Рассчитать полную поверхность треугольной пирамиды, заданной координатами своих вершин. Для определения расстояния между точками в пространстве оформить функцию.
8. Дано число n . Получить все простые делители этого числа. Процедуру распознавания простого числа оформить отдельно. (Напомним, что простым называется число, которое не имеет целочисленных делителей, кроме единицы и самого себя).

9. Составить программу, которая число, заданное в десятичной системе счисления, переведет в: а) двоичную систему счисления; б) восьмеричную. Перевод в каждую из систем счисления оформить отдельной функцией.
10. Даны координаты вершин треугольника на плоскости и координаты точки внутри него. Найти минимальное расстояние от этой точки до стороны треугольника. Оформить вычисление расстояния от точки до прямой отдельной функцией.
11. Найти высоты треугольника, заданного координатами своих вершин. Указать наименьшую из них. Для определения стороны треугольника оформить отдельную функцию.
12. Написать функцию, которая вычисляет сопротивление цепи, состоящей из трех проводников. Параметрами ее являются значения сопротивлений, а также тип соединения – последовательное или параллельное (цифрами 1 или 2). Проверить ее в работе, написав программу с ее использованием.
13. Написать программу для вычисления косинуса угла между векторами, заданными своими координатами (скалярное произведение векторов делится на произведение модулей этих векторов). Для вычисления скалярного произведения и модуля вектора оформить отдельную функцию.
14. Написать функцию, которая вычисляет доход по вкладу. Исходными данными для функции являются: величина вклада, процентная ставка (годовых) и срок вклада (количество дней). Проверить ее в работе, написав программу с ее использованием.
15. Написать функцию, которая возвращает преобразованную к верхнему регистру строку, полученную в качестве аргумента (т.е. вместо малых букв выводит строку заглавными буквами). Функция должна работать как для латинских, так и для русских букв. Проверить ее в работе, написав программу с ее использованием.
16. Написать функцию решения квадратного уравнения. Исходными данными для подпрограммы должны быть коэффициенты уравнения, а выдавать подпрограмма должна сообщение о том, есть корни или нет корней, их количество и их значение. Проверять вводимые данные (коэффициент при x^2 не должен быть нулем). Проверить ее в работе, написав программу с ее использованием.
17. Даны четыре натуральных числа. Найти наибольший общий делитель (НОД) для этих четырех чисел. Поиск НОД двух чисел оформить функцией.
18. Дано натуральное число n . Среди чисел $1, 2, \dots, n$ найти все те, которые можно представить в виде суммы квадратов двух натуральных чисел. (Определить функцию, позволяющую распознавать полные квадраты.)
19. Даны действительные числа $x_1, y_1, x_2, y_2, \dots, x_{10}, y_{10}$. Найти периметр десятиугольника, вершины которого имеют соответственно координаты $(x_1, y_1), (x_2, y_2), \dots, (x_{10}, y_{10})$. (Определить функцию вычисления расстояния между двумя точками, заданными своими координатами.)
20. Дано натуральное число n . Выяснить, имеются ли среди чисел $n, n+1, \dots, 2n$ близнецы, т.е. простые числа, разность между которыми равна двум. (Определить процедуру, позволяющую распознавать простые числа.) В пяти следующих задачах потребуются сократить обыкновенную дробь результата. Процедуру сокращения дроби оформить в виде отдельной функции (напомним, что для этого нужно искать наибольший общий делитель для числителя и знаменателя дроби).
21. Составить программу для сложения двух обыкновенных дробей $a/b + c/d$.
22. Составить программу для вычитания двух обыкновенных дробей $a/b - c/d$.
23. Составить программу для умножения двух обыкновенных дробей $a/b * c/d$.
24. Составить программу для деления двух обыкновенных дробей $a/b : c/d$.
25. Составить программу для возведения в степень обыкновенной дроби.

Лабораторная работа 5. Структуры, объединения и перечисления.

В задачах 1 – 4 одинаковое начало условия задачи:

Дан список учащихся из 10 записей. Каждая запись имеет поля: фамилия, имя, номер класса, буква класса.

1. Найти однофамильцев, обучающихся в одном классе.
2. Найти тезок (одинаковые имена), обучающихся в одном классе.
3. Найти двух учащихся, у которых совпадают имя и фамилия.
4. Вывести фамилию и первую букву имени для всех учеников указанного извне класса.

В задачах 5 – 8 одинаковое начало условия задачи

Багаж пассажира характеризуется количеством вещей (целый тип) и общим весом вещей (вещественный тип). Дан список из сведений о багаже 10 пассажиров.

5. Найти багаж, средний вес одной вещи, в котором отличается не более, чем на 0,3 кг от общего среднего веса одной вещи по всему списку.
6. Найти число пассажиров, имеющих более двух вещей.
7. Число пассажиров, количество вещей которых превосходит среднее число вещей по всему списку.
8. Выяснить, имеется ли пассажир, багаж которого состоит из одной вещи весом менее 30 кг.

В задачах 9–10 одинаковое начало условия задачи:

Список книг состоит из 10 записей. Запись содержит поля: фамилия автора (тип string), название книги (тип строка), год издания (тип целый).

9. Найти названия книг данного автора, изданных с 1960 г.
10. Определить, имеются ли книги с названием «Информатика» и если да, то сообщить фамилии авторов, год издания этих книг.

В задачах 11–16 одинаковое начало условия задачи:

Дана структура, задающая дату вида: `Struct date {int day; int month; int year;}`; Пользуясь таким структурным типом, составить программу, определяющую:

11. дату следующего (относительно сегодняшнего) дня;
12. дату предыдущего дня;
13. дату, которая наступит через *m* дней;
14. дату, которая была за *m* дней до сегодняшнего дня;
15. число суток, прошедших от заданной даты *d1* до даты *d2*;
16. день недели, выпадающий на дату *d1*, если известно, что в первый день нашей эры был понедельник.

В задачах 17–18 одинаковое начало условия задачи:

Дана структура, задающая время, вида: `struct time { int hour; int min; int sec;}`; Пользуясь таким структурным типом, составить программу:

17. определяющую предшествует ли время *t1* времени *t2* (в пределах суток);
18. присваивающую параметру *t1* время, на 1 сек большее времени *t2* (учесть смену суток).

В задачах 19–20 одинаковое начало условия задачи:

Дан список учащихся из 10 записей. Каждая запись имеет поля: фамилия, имя, номер класса (только 8-е и 10-е классы):

19. вывести отдельно учеников 10-х классов;
20. выяснить, на сколько человек в 8-х классах больше, чем в 9-х.

В задачах 21–25 одинаковое начало условия задачи:

Дан список учащихся из 10 записей. Каждая запись имеет поля: фамилия, имя, номер класса, оценки по трем предметам:

21. вывести отдельно отличников для указанного извне класса;
22. вывести отдельно двоечников (хотя бы одна двойка) для указанного извне класса;
23. вывести отдельно «хорошистов» (оценки 4 и 5) для указанного извне класса;
24. вывести отдельно «троечников» (имеется хотя бы одна тройка, но нет двоек) для указанного извне класса;
25. вывести список всех учеников с указанием среднего балла для каждого.

4 семестр:

Тема 6. Классы и объекты

Тема 7. Конструкторы и деструкторы

Тема 8. Перегрузка операторов

Тема 9. Наследование и виртуальные функции

Тема 10. Шаблоны

Лабораторная работа 6. Классы и объекты в C++

Цель. Получить практические навыки реализации классов на C++.

Основное содержание работы.

Написать программу, в которой создаются и разрушаются объекты, определенного пользователем класса. Выполнить исследование вызовов конструкторов и деструкторов.

Порядок выполнения работы.

1. Определить пользовательский класс в соответствии с вариантом задания (смотри приложение).
2. Определить в классе следующие конструкторы: без параметров, с параметрами, копирования.
3. Определить в классе деструктор.
4. Определить в классе компоненты-функции для просмотра и установки полей данных.
5. Определить указатель на компоненту-функцию.
6. Определить указатель на экземпляр класса.
7. Написать демонстрационную программу, в которой создаются и разрушаются объекты пользовательского класса и каждый вызов конструктора и деструктора сопровождается выдачей соответствующего сообщения (какой объект какой конструктор или деструктор вызвал).
8. Показать в программе использование указателя на объект и указателя на компоненту-функцию.

Методические указания.

1. Пример определения класса.

```
const int LNAME=25;
class STUDENT{
char name[LNAME];      // имя
int age;                // возраст
float grade;           // рейтинг
```

```

public:
STUDENT();           // конструктор без параметров
STUDENT(char*,int,float); // конструктор с параметрами
STUDENT(const STUDENT&); // конструктор копирования
~STUDENT();
char * GetName() ;
int GetAge() const;
float GetGrade() const;
void SetName(char*);
void SetAge(int);
void SetGrade(float);
void Set(char*,int,float);
void Show(); };

```

Более профессионально определение поля name типа указатель:

char* name. Однако в этом случае реализация компонентов-функций усложняется.

2. Пример реализации конструктора с выдачей сообщения.

```

STUDENT::STUDENT(char*NAME,int AGE,float GRADE)
{
strcpy(name,NAME); age=AGE; grade=GRADE;
cout<< "\nКонструктор с параметрами вызван для объекта
<<this<<endl;
}

```

3. Следует предусмотреть в программе все возможные способы вызова конструктора копирования. Напоминаем, что конструктор копирования вызывается:

а) при использовании объекта для инициализации другого объекта

Пример.

```
STUDENT a("Иванов",19,50), b=a;
```

б) когда объект передается функции по значению

Пример.

```
void View(STUDENT a){a.Show;}
```

в) при построении временного объекта как возвращаемого значения функции

Пример.

```

STUDENT NoName(STUDENT & student)
{STUDENT temp(student);
temp.SetName("NoName");
return temp;}

```

```
STUDENT c=NoName(a);
```

4. В программе необходимо предусмотреть размещение объектов как в статической, так и в динамической памяти, а также создание массивов объектов.

Примеры.

а) массив студентов размещается в статической памяти

```

STUDENT группа[3];
группа[0].Set("Иванов",19,50);
и т.д.

```

или

```

STUDENT группа[3]={STUDENT("Иванов",19,50),
STUDENT("Петрова",18,25.5),
STUDENT("Сидоров",18,45.5)};

```

б) массив студентов размещается в динамической памяти

```

STUDENT *p;
p=new STUDENT [3];
p-> Set("Иванов",19,50);
и т.д.

```

5. Пример использования указателя на компонентную функцию

```

void (STUDENT::*pf)();
pf=&STUDENT::Show;
(p[1].*pf)();

```

6. Программа использует три файла:

- заголовочный h-файл с определением класса,
- сpp-файл с реализацией класса,
- сpp-файл демонстрационной программы.

Для предотвращения многократного включения файла-заголовка следует использовать директивы препроцессора

```
#ifndef STUDENTH
#define STUDENTH
// модуль STUDENT.H
...
#endif
```

Содержание отчета.

1. Титульный лист: название дисциплины; номер и наименование работы; фамилия, имя, отчество студента; дата выполнения.
2. Постановка задачи. Следует дать конкретную постановку, т.е. указать, какой класс должен быть реализован, какие должны быть в нем конструкторы, компоненты-функции и т.д.
3. Определение пользовательского класса с комментариями.
4. Реализация конструкторов и деструктора.
5. Фрагмент программы, показывающий использование указателя на объект и указателя на функцию с объяснением.
6. Листинг основной программы, в котором должно быть указано, в каком месте и какой конструктор или деструктор вызываются.

Приложение. Варианты заданий.

Описания членов - данных пользовательских классов

- | | | |
|-------------------|--------------------|----------------------|
| 1. СТУДЕНТ | | |
| имя – char* | 2. СЛУЖАЩИЙ | |
| курс – int | имя – char* | 3. КАДРЫ |
| пол – int(bool) | возраст – int | имя – char* |
| | рабочий стаж – int | номер цеха – int |
| 4. ИЗДЕЛИЕ | | разряд – int |
| имя – char* | 5. БИБЛИОТЕКА | |
| шифр – char* | имя – char* | 6. ЭКЗАМЕН |
| количество – int | автор – char* | имя студента – char* |
| | стоимость – float | дата – int |
| 7. АДРЕС | | оценка – int |
| имя – char* | 8. ТОВАР | |
| улица – char* | имя – char* | 9. КВИТАНЦИЯ |
| номер дома – int | количество – int | номер – int |
| | стоимость – float | дата – int |
| 10. ЦЕХ | | сумма – float |
| имя – char* | 11. ПЕРСОНА | |
| начальник – char* | имя – char* | 12. АВТОМОБИЛЬ |
| количество | возраст – int | марка – char* |
| работающих – int | пол – int(bool) | мощность – int |
| | стоимость – float | |
| 13. СТРАНА | | |
| имя – char* | 14. ЖИВОТНОЕ | |
| форма | имя – char* | 15. КОРАБЛЬ |
| правления – char* | класс – char* | имя – char* |
| площадь – float | средний вес – int | водоизмещение – int |
| | тип – char* | |

Лабораторная работа 7. Конструкторы и деструкторы.

Цели работы:

1. Освоить на практике создание пользовательских типов.

2. Выяснить назначение и принципы работы конструкторов и деструкторов, а также экспериментально определить ситуации когда они вызываются.
3. Выяснить влияние спецификатора доступа на доступность полей и методов внутри и извне класса.

Задание:

1. Создать класс "Студент" содержащую следующие поля:

- имя студента
- отчество студента
- фамилию студента
- год рождения
- группа

2. Определить конструктор для инициализации полей по умолчанию. Определить конструктор с параметрами и деструктор. Написать тестовый пример.

3. Добавить в начале класса спецификатор доступа private. Запустить программу. Какие возникли проблемы? В каких именно местах программы и к чему запрещен доступ? Почему? Как исправить возникшее положение?

4. Написать интерфейсные функции доступа к полям класса (получить/задать значение поля), вынося поля в секцию private, разрешив доступ к конструкторам, деструкторам и интерфейсным функциям.

5. Провести те же действия, что и в задании 4, но изменив private на protected. В чем разница между ними?

6. Внести в конструкторы и деструктор выдачу сообщений на экран о том, какая функция была вызвана. В начале и конце основной программы поставить выдачу сообщений о начале и конце программы. Выяснить когда происходит вызовов конструкторов и деструкторов.

7. Добавить в класс свойства для инкапсуляции полей.

8. Результаты лабораторной работы оформить в виде протокола.

Контрольные вопросы:

1. Назначение конструкторов и деструкторов.
2. Когда вызываются конструкторы и деструкторы. Почему сделано таким образом?
3. В чем отличия между спецификаторами доступа private, protected, public если класс не наследуется?
4. Зачем нужны интерфейсные функции?
5. Назначения и принцип функционирования свойств.

Лабораторная работа 8. Программирование на C++ с использованием классов. Перегрузка операторов

Цель работы: Знакомство с объектно-ориентированным программированием в C++. Изучение общих понятий о классах - поля класса, методы класса (конструктор, деструктор и другие методы) [1, с. 264-289].

Изучение возможности перегрузки операторов в C++ [1, с. 294-303].

Задание. Согласно варианту (табл. 16.1) задания написать программу на языке C++.

Таблица 16.1. Варианты исходных данных для лабораторной работы

Вариант 1			
Класс	Члены класса	Методы	Операторы перегрузки
Комплексное число в алгебраической форме $a = x + y \cdot i$	Действительная x и мнимая y части числа	Конструктор, деструктор, метод вычисления модуля комплексного числа и $\sqrt[n]{a}$, метод вывода комплексного числа	Сложение (+), вычитание (-), умножение (*), деление (/) комплексных чисел, увеличение на 1 (++) действительной, мнимой части
Исходные данные		Результаты	
a, b, c, d – комплексные числа		$R = \sqrt[3]{a - \frac{b+c}{a} + b \cdot d}$. Найти модуль числа R . Увеличить на 1 действительную и мнимую часть R	
Вариант 2			
Класс	Члены класса	Методы	Операторы перегрузки
Обыкновенная дробь	Числитель и знаменатель	Конструктор, деструктор, метод сокращения дроби, метод вывода дроби	Сложение (+), вычитание (-), умножение (*), деление (/) дробей
Исходные данные		Результаты	
Даны дроби $\frac{a}{b}, \frac{c}{d}, \frac{e}{f}, \frac{g}{h}, \frac{k}{l}$		$Z = \frac{a+c}{b \cdot d} \cdot \left(\frac{g}{h} - \frac{k}{l}\right)$ На экран вывести несокращенную и сокращенную дробь Z	

Вариант 3			
Класс	Члены класса	Методы	Операторы перегрузки
Вектор	3 прямоугольн ые декартовы е координаты	Конструктор, деструктор, метод вывода вектора, метод вычисления длины вектора	Сложение (+), скалярное (%) и векторное (*) произведения векторов
Исходные данные		Результаты	
$a=\{a_x, a_y, a_z\}$ $b=\{b_x, b_y, b_z\}$ $c=\{c_x, c_y, c_z\}$		$r=(a+b) \times c;$ $t=(a+c)c$ Найти длины исходных и результирующего векторов	
Вариант 4			
Класс	Члены класса	Методы	Операторы перегрузки
Матрица	Размерность матрицы, элементы матрицы	Конструктор, деструктор, метод вывода матрицы, метод вычисления определителя матрицы	Сложение (+), вычитание (-), умножение (*) 2-х матриц, умножение матрицы на число
Исходные данные		Результаты	
$A = \begin{pmatrix} 5 & 1 & 7 \\ -10 & -2 & 1 \\ 0 & 1 & 2 \end{pmatrix}$ $B = \begin{pmatrix} 2 & 4 & 1 \\ 3 & 1 & 0 \\ 7 & 2 & 1 \end{pmatrix}$		$C=2(A-B)(A^2 + B)$ Найти C	
Вариант 5			
Класс	Члены класса	Методы	Операторы перегрузки
Прямая	Координаты двух точек (x1,y1) и (x2,y2)	Конструктор, деструктор, метод вывода уравнения прямой	Проверка параллельности 2-х прямых (), определение угла между 2-мя прямыми (%)
Исходные данные		Результаты	
Прямая АВ: $A(xa, ya), B(xb, yb)$ Прямая CD: $C(xc, yc), D(xd, yd)$		Вывести уравнения прямых линий. Если прямые не параллельны, то найти и вывести угол между ними	

Лабораторная работа 9. Наследование. Виртуальные функции. Полиморфизм

Цель задания:

- 1) Создание консольного приложения, состоящего из нескольких файлов в системе программирования Visual Studio.
- 2) Создание иерархии классов с использованием простого наследования и абстрактного класса.
- 3) Изучение полиморфизма и виртуальных методов.

Ход работы

Задача

Базовый класс:

МАШИНА

торговая_марка - string

число_цилиндров - int

мощность - int

Создать производный класс ГРУЗОВИК, добавив в него характеристику грузоподъемности кузова типа int.

1. Создать пустой проект.
2. Добавить в проект класс Object.
3. В файл Object.h ввести следующее описание абстрактного класса Object:

`#pragma once`

`class Object`

`{`

`public:`

`Object(void); public:`

`~Object(void);`

`virtual void Show()=0;//чисто виртуальная функция`

```
};
```

4. Файл Object.cpp оставить без изменения.
5. Добавить в проект класс Car, унаследованный от класса Object. В классе переопределить метод Show().
6. В файле Car.h должно быть следующее описание класса

```
#pragma once
#include "object.h" #include
<string> #include
<iostream> using namespace
std; class Car:
    public Object
{
public:
    Car(void); public:
    virtual ~Car(void);
    void Show();//функция для просмотра атрибутов класса с помощью указателя Car(string,int,int);
    Car(const Car&);
    string Get_mark(){return mark;} int
    Get_cyl(){return cyl;}
    int Get_power(){return power;} void
    Set_mark(string);
    void Set_cyl(int); void
    Set_power(int);
    Car& operator=(const Car&);
    friend istream& operator>>(istream&in,Car&c); friend ostream&
    operator<<(ostream&out,const Car&c);
protected:
    string mark; int
    cyl;
    int power;
};
```

7. В файле Car.cpp должно быть следующее определение методов класса

```
#include "Car.h"

Car::Car(void)
{
    mark=""
    ";
    cyl=0;
    power=
    0;
}
Car::~Car(void)
{
}
Car::Car(string M,int C,int P)
{
    mark=
    M;
    cyl=C;
    power=
    P;
}
Car::Car(const Car& car)
{
    mark=car.mark;
    cyl=car.cyl;
    power=car.power;
}
void Car::Set_cyl(int C)
```

```

{
    cyl=C;
}
void Car::Set_mark(string M)
{
    mark=M;
}
void Car::Set_power(int P)
{
    power=P;
}
Car& Car::operator=(const Car&c)
{
    if(&c==this)return *this;
    mark=c.mark; power=c.power;
    cyl=c.cyl;
    return *this;
}
istream& operator>>(istream&in,Car&c)
{
    cout<<"\nMark: "; in>>c.mark;
    cout<<"\nPower: ";in>>c.power;
    cout<<"\nCyl: ";in>>c.cyl; return in;
}
ostream& operator<<(ostream&out,const Car&c)
{
    out<<"\nMARK : "<<c.mark;
    out<<"\nCYL : "<<c.cyl;
    out<<"\nPOWER : "<<c.power;
    out<<"\n";
    return out;
}
void Car::Show()
{
    cout<<"\nMARK : "<<mark;
    cout<<"\nCYL : "<<cyl;
    cout<<"\nPOWER : "<<power;
    cout<<"\n";
}

```

8. Добавить в проект файл Lab5_main.cpp и проверить работу класса Car. Для этого нужно создать переменную класса Car, ввести и вывести значения атрибутов этой переменной:

```

#include "Object.h" #include
"Car.h" #include <string>
#include <iostream> using
namespace std; void main()
{
    Car a;
    cin>>
    a;
    cout<<a<<<endl;//вывод с помощью перегруженной операции Object *p=&a;
    p->Show();//вывод с помощью метода Show() и указателя
}

```

9. Откомпилировать и выполнить программу.
 10. Добавить в проект класс Lorry, унаследованный от класса Car. В классе переопределить метод Show().
 11. В файле Lorry.h должно быть следующее описание класса

```

#pragma once #include
"car.h" class Lorry :
    public Car
{
public:

```

```

Lorry(void); public:
    ~Lorry(void);
    void Show();//функция для просмотра атрибутов класса с помощью указателя Lorry(string,int,int,int);
    Lorry(const Lorry & );
    int Get_gruz(){return груз;} void
    Set_Gruz(int);
    Lorry& operator=(const Lorry&);
    friend istream& operator>>(istream&in,Lorry&l); friend ostream&
    operator<<(ostream&out,const Lorry&l);
protected:
    int груз;
};

```

12. В файле Lorry.cpp должно быть следующее описание класса

```

#include "Lorry.h"
Lorry::Lorry(void):Car()
{
    груз=0;
}
Lorry::~Lorry(void)
{
}
Lorry::Lorry(string M, int C, int P, int G):Car(M,C,P)
{
    груз=G;
}
Lorry::Lorry(const Lorry &L)
{
    mark=L.mark;
    cyl=L.cyl;
    power=L.power;
    груз=L.груз;
}
void Lorry::Set_Gruz(int G)
{
    груз=G;
}
Lorry& Lorry::operator=(const Lorry&l)
{
    if(&l==this)return *this;
    mark=l.mark; power=l.power;
    cyl=l.cyl;
    return *this;
}
istream& operator>>(istream&in,Lorry&l)
{
    cout<<"\nMark:"; in>>l.mark;
    cout<<"\nPower:";in>>l.power;
    cout<<"\nCyl:";in>>l.cyl;
    cout<<"\nGruz:";in>>l.груз; return in;
}
ostream& operator<<(ostream&out,const Lorry&l)
{
    out<<"\nMARK : "<<l.mark;
    out<<"\nCYL : "<<l.cyl;
    out<<"\nPOWER : "<<l.power;
    out<<"\nGRUZ : "<<l.груз;
    out<<"\n";
    return out;
}
void Lorry::Show()
{

```

```

    cout<<"\nMARK : "<<mark;
    cout<<"\nCYL : "<<cyl;
    cout<<"\nPOWER : "<<power;
    cout<<"\nGRUZ : "<<gruz;
    cout<<"\n";
}

```

13. Проверить работу класса Lorry. Для этого нужно создать переменную класса Lorry, ввести и вывести значения атрибутов этой переменной:

```

#include "Object.h" #include
"Car.h" #include <string>
#include <iostream> using
namespace std; void main()
{
.....
    Lorry b;
    cin>>b;
    cout<<b<<endl;//вывод с помощью перегруженной операции p=&b;
    p->Show();//вывод с помощью метода Show() и указателя
}

```

14. Откомпилировать и выполнить программу.

15. Добавить в проект класс Vector, который будет содержать указатели на объекты абстрактного класса Object.

16. В файл Vector.h ввести следующее описание класса Vector:

```

#pragma once
#include "object.h"
#include <string> #include
<iostream>

using namespace std; class
Vector
{
public:
    Vector(void);//конструктор без параметров
    Vector(int);//конструктор копирования
public:
    ~Vector(void);//деструктор
    void Add(Object *);//добавление элемента в вектор
friend ostream& operator<<(ostream&out,const Vector&);//операция вывода private:
    Object**beg;//указатель на первый элемент вектора int size;//размер
    int cur;//текущая позиция
};

```

17. В файл Vector.cpp ввести следующее определение методов класса Vector:

```

#include "Vector.h"
//конструктор без параметров Vector::Vector(void)
{
    beg=0
    ;
    size=0
    ;
    cur=0;
}
//деструктор
Vector::~Vector(void)
{
    if(beg!=0)delete [] beg; beg=0;
}
//конструктор с параметрами Vector::Vector(int n)
{
    beg=new Object*[n];
    cur=0;
}

```

```

        size=n;
    }
    //добавление объекта, на который указывает указатель p в вектор void Vector::Add(Object
    *p)
    {
        if(cur<size)
        {
            beg[cur]=p;
            cur++;
        }
    }
    //операция вывода
    ostream& operator<<(ostream&out,const Vector&v)
    {
        if(v.size==0) out<<"Empty"<<endl;
        Object **p=v.beg;//указатель на указатель типа Object for(int i=0;i<v.cur;i++)
        {

        }

        return out;
    }
    (*p)->Show();//вызов метода Show() (позднее связывание) p++;//передвигаем указатель на следующий объект
    18. Проверить работу класса Vector. Для этого нужно создать переменную класса Vector, добавить в
    него объекты разных классов из построенной иерархии и вывести значения элементов вектора.
#include "Object.h" #include
"Car.h" #include "Lorry.h"
#include "Vector.h" #include
<string> #include
<iostream> using namespace
std; void main()
{
    Vector v(5);//вектор из 5 элементов Car a;//объект
    класса Car
    cin>>a;
    Lorry b;// объект класса Lorry cin>>b;
    Object*p=&a;//ставим указатель на объект класса Car
    v.Add(p);//добавляем объект в вектор
    p=&b;//ставим указатель на объект класса Lorry v.Add(p);
    //добавляем объект в вектор cout<<v;//вывод вектора
}

```

Варианты

№	Задание
1	<p>Базовый класс: ПАРА_ЧИСЕЛ (PAIR) Первое_число (first) - int Второе_число (second) – int Определить методы изменения полей и сравнения пар (пара p1 больше пары p2, если (p1.first>p2.first) (p1.first==p2.first && p1.second>p2.second)). Создать производный класс ДРОБЬ (FRACTION), с полями Целая_часть_числа и Дробная_часть_числа. Определить полный набор методов сравнения.</p>

2	<p>Базовый класс: ПАРА_ЧИСЕЛ (PAIR) Первое_число (first) - int Второе_число (second) – int Определить методы изменения полей и вычисления произведения чисел. Создать производный класс ПРЯМОУГОЛЬНИК (RECTANGLE), с полями- сторонами. Определить методы для вычисления площади и периметра прямоугольника.</p>
3	<p>Базовый класс: ПАРА_ЧИСЕЛ (PAIR) Первое_число (first) - int Второе_число (second) – int Определить методы изменения полей и вычисления произведения чисел. Создать производный класс ПРЯМОУГОЛЬНЫЙ_ТРЕУГОЛЬНИК (RIGHTANGLED), с полями- катетами. Определить метод вычисления гипотенузы.</p>
4	<p>Базовый класс: ПАРА_ЧИСЕЛ (PAIR) Первое_число (first) - int Второе_число (second) – int Определить методы изменения полей и операцию сложения пар $(a,b)+(c,d)=(a+b,c+d)$ Создать производный класс КОМПЛЕКСНОЕ_ЧИСЛО(COMPLEX), с полями Действительная_часть_числа и Мнимая_часть_числа. Определить операции умножения $(a,b)*(c,d)= (a*c-b*d, a*d+b*c)$ и вычитания $(a,b)-(c,d)= (a-b, c-d)$</p>
5	<p>Базовый класс: ПАРА_ЧИСЕЛ (PAIR) Первое_число (first) - int Второе_число (second) – int Определить методы изменения полей и операцию сложения пар $(a,b)+(c,d)=(a+b,c+d)$ Создать производный класс ДЕНЕЖНАЯ_СУММА(MONEY), с полями Рубли и Копейки. Переопределить операцию сложения и определить операции вычитания и деления денежных сумм.</p>
6	<p>Базовый класс: ПАРА_ЧИСЕЛ (PAIR) Первое_число (first) - int Второе_число (second) – int Определить методы изменения полей и операцию сложения пар $(a,b)+(c,d)=(a+b,c+d)$ Создать производный класс ДЛИННОЕ_ЧИСЛО(LONG), с полями Старшая_часть_числа и Младшая_часть_числа. Переопределить операцию сложения и определить операции вычитания и умножения.</p>
7	<p>Базовый класс: ПАРА_ЧИСЕЛ (PAIR) Первое_число (first) - int Второе_число (second) – int Определить методы проверки на равенство и операцию перемножения полей. Реализовать операцию вычитания пар по формуле $(a,b)-(c,d)=(a-b,c-d)$ Создать производный класс ПРОСТАЯ_ДРОБЬ(RATIONAL), с полями Числитель и Знаменатель. Переопределить операцию вычитания и определить операции сложения и умножения простых дробей.</p>

8	<p>Базовый класс: ТРОЙКА_ЧИСЕЛ (TRIAD) Первое_число (first) - int Второе_число (second) – int Третье_число (third) - int Определить методы изменения полей и сравнения триады. Создать производный класс DATE с полями год, месяц и число. Определить полный набор операций сравнения дат.</p>
9	<p>Базовый класс: ТРОЙКА_ЧИСЕЛ (TRIAD) Первое_число (first) - int Второе_число (second) – int Третье_число (third) - int Определить методы изменения полей и сравнения триады. Создать производный класс TIME с полями часы, минуты и секунды. Определить полный набор операций сравнения временных промежутков.</p>
10	<p>Базовый класс: ТРОЙКА_ЧИСЕЛ (TRIAD) Первое_число (first) - int Второе_число (second) – int Третье_число (third) - int Определить методы изменения полей и увеличения полей на 1. Создать производный класс DATE с полями год, месяц и число. Переопределить методы увеличения полей на 1 и определить метод увеличения даты на n дней.</p>
11	<p>Базовый класс: ТРОЙКА_ЧИСЕЛ (TRIAD) Первое_число (first) - int Второе_число (second) – int Третье_число (third) - int Определить методы изменения полей и увеличения полей на 1. Создать производный класс TIME с полями часы, минуты и секунды. Переопределить методы увеличения полей на 1 и определить методы увеличения на n секунд и минут.</p>
12	<p>Базовый класс: ЧЕЛОВЕК (PERSON) Имя (name) – string Возраст (age) – int Определить методы изменения полей. Создать производный класс STUDENT, имеющий поле год обучения. Определить методы изменения и увеличения года обучения.</p>
13	<p>Базовый класс: ЧЕЛОВЕК (PERSON) Имя (name) – string Возраст (age) – int Определить методы изменения полей. Создать производный класс EMPLOYEE, имеющий поля Должность – string и Оклад – double. Определить методы изменения полей и вычисления зарплаты сотрудника по формуле Оклад+Премия(% от оклада).</p>

14	<p>Базовый класс: ЧЕЛОВЕК (PERSON) Имя (name) – string Возраст (age) – int Определить методы изменения полей. Создать производный класс TEACHER, имеющий поля Предмет – string и Количество часов – int. Определить методы изменения полей, а также увеличения и уменьшения часов.</p>
15	<p>Базовый класс: ЧЕЛОВЕК (PERSON) Имя (name) – string Возраст (age) – int Определить методы изменения полей. Создать производный класс STUDENT, имеющий поля Предмет – string и Оценка – int. Определить методы изменения полей и метод, выдающий сообщение о неудовлетворительной оценке.</p>

Контрольные вопросы

1. Какой класс называется абстрактным?
2. Какой метод называется чисто виртуальным? Чем он отличается от виртуального метода?
3. Для чего предназначены абстрактные классы?
4. Что такое полиморфные функции?
5. Чем полиморфизм отличается от принципа подстановки?
6. Привести примеры иерархий с использованием абстрактных классов.
7. Привести примеры полиморфных функций.
8. В каких случаях используется механизм позднего связывания?

4.1.3. Контрольная работа

4 семестр:

Тема 1. Основы языка C++

Тема 2. Управляющие конструкции

Тема 3. Указатели, ссылки, массивы и текстовые строки

Тема 4. Функции

Тема 5. Структуры, объединения и перечисления

4 семестр:

Тема 6. Классы и объекты

Тема 7. Конструкторы и деструкторы

Тема 8. Перегрузка операторов

Тема 9. Наследование и виртуальные функции

Тема 10. Шаблоны

4.1.3.1. Порядок проведения.

Контрольная работа проводится в часы аудиторной работы. Обучающиеся получают задания для проверки усвоения пройденного материала. Работа выполняется в письменном виде и сдаётся преподавателю. Оцениваются владение материалом по теме работы, аналитические способности, владение методами, умения и навыки, необходимые для выполнения заданий

4.1.3.2 Критерии оценивания

9-10 баллов ставится, если обучающийся:

Правильно выполнил все задания. Продемонстрировал высокий уровень владения материалом. Проявлены превосходные способности применять знания и умения к выполнению конкретных заданий. .

7-8 баллов ставится, если обучающийся:

Правильно выполнил большую часть заданий. Присутствуют незначительные ошибки. Продемонстрирован хороший уровень владения материалом. Проявлены средние способности применять знания и умения к выполнению конкретных заданий.

5-6 баллов ставится, если обучающийся:

Задания выполнил более чем наполовину. Присутствуют серьёзные ошибки. Продемонстрирован удовлетворительный уровень владения материалом. Проявлены низкие способности применять знания и умения к выполнению конкретных заданий.

0-4 балла ставится, если обучающийся:

Задания выполнил менее чем наполовину. Продемонстрировал неудовлетворительный уровень владения материалом. Проявлены недостаточные способности применять знания и умения к выполнению конкретных заданий.

4.1.3.3. Содержание оценочного средства

Тема 1. Основы языка C++

Тема 2. Управляющие конструкции

Тема 3. Указатели, ссылки, массивы и текстовые строки

Тема 4. Функции

Тема 5. Структуры, объединения и перечисления

Задания:

1. Написать программу для вычисления факториала числа n . Число n вводится пользователем с клавиатуры.
2. Написать программу для поиска наименьшего элемента массива. Массив заполнить случайными числами.
3. Написать программу для вычисления скалярного произведения двух векторов. Векторы реализуются в виде массивов, их названия передаются аргументами функции.
4. Написать программу для реализации комплексных чисел в алгебраической форме в виде элементов структуры. Предусмотреть функции для сложения, вычитания, деления и умножения комплексных чисел.

Тема 6. Классы и объекты

Тема 7. Конструкторы и деструкторы

Тема 8. Перегрузка операторов

Тема 9. Наследование и виртуальные функции

Задания:

1. Написать программу для вычисления модуля вектора в трехмерном декартовом пространстве. Вектор реализовать в виде объекта класса, поля которого являются элементами (координатами) вектора. Для вычисления модуля вектора создать специальный метод.
2. Написать программу для создания, на основе конструктора, триарного дерева объектов. В такой структуре каждый объект ссылается на три объекта такого же типа. Каждый из этих объектов, в свою очередь, ссылается на три объекта и т.д.
3. Написать программу с классами для реализации полиномов степени n и $2n$. Путем перегрузки операторов предусмотреть возможность преобразования (точнее, создания на основе) объекта для полинома степени n в объект для полинома степени $2n$. Реализовать процесс вычисления произведения двух полиномов степени n (результат полином степени $2n$).
4. Написать программу с классом для реализации комплексных чисел. Предусмотреть возможность складывать, вычитать и умножать комплексные числа. Создать производный класс для реализации квадратных матриц с комплексными элементами. Элементы матрицы заносятся в двумерный массив. Предусмотреть возможность индексирования объектов и умножения матриц.

4.2. Оценочные средства промежуточной аттестации

По дисциплине предусмотрен экзамен. Экзамен проходит по билетам. В каждом билете один теоретический вопрос и одно практическое задание. Экзамен проводится в устной / письменной и компьютерной форме. Оценивается владение материалом, его системное освоение, способность применять нужные знания, навыки и умения при анализе проблемных ситуаций и решении практических заданий.

4.2.1. Устный или письменный ответ на вопрос

4.2.1.1. Порядок проведения и процедура оценивания.

Устный или письменный ответ на вопрос направлен на проверку знаний основных разделов по дисциплине «Программирование».

4.2.1.2. Критерии оценивания.

17-20 баллов ставится, если обучающийся:

В ответе качественно раскрыл содержание темы. Ответ хорошо структурирован. Прекрасно освоено понятийный аппарат. Продемонстрирован высокий уровень понимания материала. Превосходное умение формулировать свои мысли, обсуждать дискуссионные положения.

14-16 баллов ставится, если обучающийся:

Основные вопросы темы раскрыл. Структура ответа в целом адекватна теме. Хорошо освоено понятийный аппарат. Продемонстрирован хороший уровень понимания материала. Хорошее умение формулировать свои мысли, обсуждать дискуссионные положения.

11-13 баллов ставится, если обучающийся:

Тему частично раскрыл. Ответ слабо структурирован. Понятийный аппарат освоен частично. Понимание отдельных положений из материала по теме. Удовлетворительное умение формулировать свои мысли, обсуждать дискуссионные положения.

0-10 баллов ставится, если обучающийся:

Тему не раскрыл. Понятийный аппарат освоен неудовлетворительно. Понимание материала фрагментарное или отсутствует. Неумение формулировать свои мысли, обсуждать дискуссионные положения.

4.2.1.3. Оценочные средства.**Вопросы для устного или письменного ответа**

1. Структурное программирование в C++.
2. Базовые типы данных.
3. Константы и литералы.
4. Арифметические операторы.
5. Логические операторы.
6. Операторы сравнения.
7. Оператор присваивания и приведение типов.
8. Тернарный оператор.
9. Побитовые операторы и двоичное представление чисел.
10. Управляющие инструкции. Ветвления и циклы.
11. Условный оператор IF().
12. Вложенные условные операторы.
13. Условный оператор SWITCH().
14. Оператор цикла FOR().
15. Оператор цикла WHILE().
16. Инструкция безусловного перехода.
17. Объявление и использование функций.
18. Передача указателя аргументом функции.
19. Передача массива аргументом функции.
20. Передача строки аргументом функции.
21. Аргументы функции MAIN().
22. Возвращение функцией указателя.
23. Возвращение функцией ссылки.
24. Указатели на функции.
25. Рекурсия.
26. Перегрузка функций.
27. Объявление и использование указателей.
28. Адресная арифметика и сравнение указателей.
29. Многоуровневая адресация.
30. Статические одномерные массивы.
31. Указатель на массив.
32. Двумерные массивы.
33. Инициализация массивов.
34. Создание и инициализация строк.
35. Динамические массивы.
36. Структуры. Массивы структур.
37. Указатели на структуры.
38. Объединения.
39. Перечисления и определение типов.
40. Объектно-ориентированное программирование.
41. Объявление класса.
42. Открытые и закрытые члены класса.
43. Статические члены класса.
44. Перегрузка методов.
45. Конструкторы. Создание и перегрузка конструктора.
46. Деструкторы. Правила создания деструктора. Использование деструкторов.
47. Внешняя операторная функция для переопределения бинарного оператора.
48. Перегрузка операторной функции.
49. Переопределение унарных операторов внешними функциями.
50. Перегрузка операторов методами класса.
51. Наследование классов и типы наследования.
52. Переопределение методов и виртуальные функции.

53. Многоуровневое наследование.
 54. Чисто виртуальные методы и абстрактные классы.
 55. Обобщённые функции. Перегрузка обобщённых функций.

4.2.2. Практическое задание

4.2.2.1. Порядок проведения.

Предлагаются задания на проверку практических навыков по дисциплине «Программирование».

4.2.2.2. Критерии оценивания.

26-30 баллов ставится, если обучающимся:

Задание выполнено полностью и правильно.

21-25 баллов ставится, если обучающимся:

Задание выполнено полностью, но нет достаточного обоснования. Или при верном решении допущена ошибка или недочет, не влияющий на правильную последовательность рассуждений.

17-20 баллов ставится, если обучающимся:

Задание выполнено частично или с фактическими ошибками.

0-16 баллов ставится, если обучающимся:

Задание не выполнено или выполнено с большим количеством фактических ошибок.

4.2.2.3. Оценочные средства.

1. Напишите программу вычисления суммы

$$S = \sum_{k=1}^{20} \frac{1}{k!(2k+1)}$$

Вычисление факториала оформите подпрограммой.

2. Напишите программу вычисления суммы

$$S = \sum_{n=1}^{10} \frac{n!}{(n+1)!(2n-1)}$$

Вычисление факториала оформите подпрограммой.

3. Напишите подпрограмму, определяющую, принадлежит ли точка с координатами (x,y) кругу радиуса r с центром в точке (a,b).
 4. Напишите программу вычисления суммы

$$1! + 2! + 3! + \dots + n!$$

используя подпрограмму вычисления факториала.

5. Напишите подпрограмму вычисления корней квадратного уравнения. Программа должна учитывать все возможные сочетания нулевых и ненулевых значений коэффициентов a, b, c.
 6. Написать программу поиска наибольшего общего делителя (НОД) двух положительных целых чисел.

Примечание: НОД ищется по алгоритму Евклида:

Пока x и y не равны 0, выполнять цикл
 Если x>y, то x=остатку от деления x на y
 Иначе y=остатку от деления y на x
 Конец цикла
 Ответ=x+y

7. В заданном наборе координат точек (x_i, y_i), i=1,2,...,5, найти две точки, расстояние между которыми минимально, и вывести на печать их номера. Для вычисления расстояния между двумя точками (x₁,y₁) и (x₂,y₂) по формуле

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

8. Напишите программу формирования массива A по правилу:

$$a_{ij} = f(i)g(j), \quad i = \overline{1,4}, \quad j = \overline{1,3},$$

где

$$f(i) = \frac{\sin i}{1+i^2}, \quad g(j) = \frac{e^j}{1+\cos j}$$

Для вычисления $f(i)$ и $g(j)$ используйте подпрограммы.

9. Дана функция

$$f(x) = \sqrt{\frac{1 + \ln(x^2 + 1)}{2,1 + x}}$$

Напишите программу формирования массива A, используя подпрограмму для вычисления $f(x)$, по правилу:

$$a_{ij} = \frac{f(i) + f(i+j)}{2ij}, \quad i = \overline{1,4}, \quad j = \overline{1,3}$$

10. Даны две функции:

$$f(i) = 0,5i^3 + 0,7i^2 - 2,1i + 0,45, \quad g(j) = \ln j$$

Напишите программу формирования массива A, используя подпрограммы для вычисления $f(i)$ и $g(j)$, по правилу:

$$a_{ij} = f(i) \frac{i-j}{f(i) - g(j)}, \quad i = \overline{1,5}, \quad j = \overline{1,4}$$

11. Файл содержит массив вещественных чисел. Определить количество чисел, меньших среднего арифметического значения всех чисел.

Пояснение: Реализовать программу обработки файлов согласно заданию. Данные читаются из текстового файла и выводятся на экран и/или в текстовый файл. Программа проверяет возможные ошибки ввода-вывода. Продемонстрировать работу программы на основе созданного теста.

12. Создать класс прямоугольников со сторонами, параллельными осям координат. Предусмотреть возможность перемещения прямоугольников на плоскости, изменение размеров, построение наименьшего прямоугольника, содержащего два заданных, и прямоугольника, являющегося общим пересечением двух прямоугольников.

13. Файл содержит фамилии студентов и их возраст. Напечатать фамилии студентов, имеющих наименьший возраст.

Пояснение: Реализовать программу обработки файлов согласно заданию. Данные читаются из текстового файла и выводятся на экран и/или в текстовый файл. Программа проверяет возможные ошибки ввода-вывода. Продемонстрировать работу программы на основе созданного теста.

14. Создать класс многочленов от одной переменной, задаваемых степенью многочлена и массивом коэффициентов. Предусмотреть методы для вычисления значения многочлена для заданного аргумента, операции сложения, вычитания и умножения многочленов с получением нового объекта-многочлена, вывод на экран описания многочлена. Написать программу, демонстрирующую работу с этим классом. Программа должна содержать меню, позволяющее осуществить проверку всех методов класса.

15. Файл содержит фамилии студентов. Вывести список студентов, имеющих самые длинные фамилии.

Пояснение: Реализовать программу обработки файлов согласно заданию. Данные читаются из текстового файла и выводятся на экран и/или в текстовый файл. Программа проверяет возможные ошибки ввода-вывода. Продемонстрировать работу программы на основе созданного теста.

16. Создать класс «домашняя библиотека». Предусмотреть возможность работы с произвольным числом книг, поиска книги по какому-либо признаку (например, по автору или по году издания), добавления книг в библиотеку, удаления книг из нее, сортировки книг по разным полям.

17. Файл содержит произвольные текстовые строки. В выходном файле все строки дополнены символом «*» до самой длинной строки.

Пояснение: Реализовать программу обработки файлов согласно заданию. Данные читаются из текстового файла и выводятся на экран и/или в текстовый файл. Программа проверяет возможные ошибки ввода-вывода. Продемонстрировать работу программы на основе созданного теста.

18. Определить класс с именем Netbook, содержащий следующие поля: название фирмы изготовителя; размер экрана; объем оперативной памяти; цена.

Реализовать методы:

- ввод данных в массив из n элементов типа Netbook;
- упорядочить по датам выпуска;
- вывод информации о товаре, размер экрана которого введен пользователем.

Перечень литературы, необходимой для освоения дисциплины (модуля)

Направление подготовки: 44.03.01 - Педагогическое образование

Профиль подготовки: Технология и робототехника

Квалификация выпускника: бакалавр

Форма обучения: очная

Язык обучения: русский

Год начала обучения по образовательной программе: 2022

Основная литература:

1. Кудрявцев, Н. Г. Элементарные основы программирования встраиваемых систем: учебное пособие / Н. Г. Кудрявцев. — Горно-Алтайск: ГАГУ, 2021. — 148 с. — Текст: электронный // Лань: электронно-библиотечная система. — URL: <https://e.lanbook.com/book/178005>
2. Мартин, Т. Микроконтроллеры ARM7. Семейство LPC2000 компании Philips. Вводный курс / Мартин Т. , пер. с англ. - М. ДОДЭКА. - 240 с. (Серия "Мировая электроника") - ISBN 978-5-94120-104-4. - Текст : электронный // ЭБС "Консультант студента": [сайт]. - URL: <https://www.studentlibrary.ru/book/ISBN9785941201044.html>
3. Хлуденев, А. В. Средства разработки и отладки программ для микроконтроллеров: учебное пособие / А. В. Хлуденев. — Оренбург: ОГУ, 2019. — 106 с. — ISBN 978-5-7410-2400-3. — Текст: электронный // Лань: электронно-библиотечная система. — URL: <https://e.lanbook.com/book/159899>
4. Сильвашко, С. А. Основы программирования микроконтроллеров на C++: учебное пособие / С. А. Сильвашко. — Оренбург: ОГУ, 2019. — 126 с. — ISBN 978-5-7410-2398-3. — Текст: электронный // Лань: электронно-библиотечная система. — URL: <https://e.lanbook.com/book/160013>
5. Смирнов, Ю. А. Основы микроэлектроники и микропроцессорной техники: учебное пособие / Ю. А. Смирнов, С. В. Соколов, Е. В. Титов. — 2-е изд., испр. — Санкт-Петербург: Лань, 2022. — 496 с. — ISBN 978-5-8114-1379-9. — Текст: электронный // Лань: электронно-библиотечная система. — URL: <https://e.lanbook.com/book/211292>

Перечень информационных технологий, используемых для освоения дисциплины (модуля), включая перечень программного обеспечения и информационных справочных систем

Направление подготовки: 44.03.01 - Педагогическое образование

Профиль подготовки: Технология и робототехника

Квалификация выпускника: бакалавр

Форма обучения: очное

Язык обучения: русский

Год начала обучения по образовательной программе: 2022

Освоение дисциплины (модуля) предполагает использование следующего программного обеспечения и информационно-справочных систем:

1. Microsoft office professional plus 2010
2. Kaspersky Endpoint Security для Windows
3. Электронная библиотечная система «ZNANIUM.COM»
4. Электронная библиотечная система Издательства «Лань»
5. Электронная библиотечная система «Консультант студента»