МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования
«Казанский (Приволжский) федеральный университет»
Елабужский институт (филиал) КФУ

УТВЕРЖДАЮ
Заместитель директора по
образовательной деятельности
_____ С.Ю. Бахвалов
«19» мая 2025 г.
МП

Программа дисциплины (модуля)
Программирование

Направление подготовки/специальность: 44.03.04 Профессиональное обучение (по отраслям)

Направленность (профиль) подготовки (специальности): Автоматизация энергетических систем

Квалификация: бакалавр

Форма обучения: заочная

Язык обучения: русский

Год начала обучения по образовательной программе: 2025

Содержание

Программу дисциплины разработал(а)(и) старший преподаватель, б/с 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
УК-1	Способен осуществлять поиск, критический анализ и синтез информации, применять системный подход для решения поставленных задач
	УК-1.1 Знать принципы поиска информации, критического анализа и синтеза информации, методики системного подхода для решения поставленных задач
	УК-1.2 Уметь осуществлять поиск, критический анализ и синтез информации, применять системный подход для решения поставленных задач
	УК-1.3 Владеть навыками поиска, критического анализа и синтеза информации; способностью применять системный подход для решения поставленных задач

Обучающийся, освоивший дисциплину (модуль):

Должен знать:

принципы эффективного поиска, критического анализа и синтеза информации по основам программирования, методики системного подхода к программированию, включая объектно-ориентированную, для решения задач на ЭВМ.

Должен уметь:

осуществлять эффективный поиск, критический анализ и синтез информации по основам программирования, использовать методики системного подхода к программированию, включая объектно-ориентированную, для решения задач на ЭВМ.;

Должен владеть:

навыками эффективного поиска, критического анализа и синтеза информации по основам программирования; способностью применять методики системного подхода к программированию, включая объектно-ориентированную, для решения задач на ЭВМ;

2. Место дисциплины (модуля) в структуре ОПОП ВО

Данная дисциплина (модуль) включена в раздел "Б1.О.04 Дисциплины (модули)" основной профессиональной образовательной программы 44.03.04 "Профессиональное обучение (по отраслям), профиль (Автоматизация энергетических систем)" и относится к дисциплинам обязательной части. Осваивается на 3 курсе установочная и зимняя сессии.

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 6 зачетных(ые) единиц(ы) на 216 часа(ов).

Контактная работа - 24 часа(ов), в том числе лекции - 10 часа(ов), практические занятия - 0 часа(ов), лабораторные работы - 14 часа(ов), контроль самостоятельной работы - 0 часа(ов).

Самостоятельная работа - 179 часа(ов).

Контроль (зачёт / экзамен) - 13 часа(ов).

Форма промежуточного контроля дисциплины: зачет 3 курс установочная сессия, экзамен 3 курс зимняя сессия.

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++	5	1	0	2	30
2.	Тема 2. Управляющие конструкции	5	1	0	2	40
3.	Тема 3. Указатели, ссылки, массивы и текстовые строки	5	2	0	4	28
4.	Тема 4. Функции	5	2	0	2	28
5.	Тема 5. Структуры, объединения и перечисления	5	2	0	2	29
6.	Тема 6. Классы и объекты	5	2	0	2	24
	Итого: 203 час и 13 час контроль		10	0	14	179

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.

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)

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 основной и дополнительной 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 осуществляющих освоение данной дисциплины (модуля).

Перечень основной и дополнительной 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	Лекционные занятия проводятся с использованием интерактивных технологий и предполагают активное участие студентов. Для подготовки к занятиям рекомендуется выделять в материале проблемные вопросы, затрагиваемые преподавателем в лекции, и группировать информацию вокруг них. Желательно выделять в используемой литературе постановки вопросов, на которые разными авторам могут быть даны различные ответы. На основании постановки таких вопросов следует собирать аргументы в пользу различных вариантов решения поставленных проблем.
лабораторные работы	Лабораторные занятия - это одна из разновидностей практического занятия, являющаяся эффективной формой учебных занятий в организации высшего образования. Лабораторные занятия имеют выраженную специфику в зависимости от учебной дисциплины, углубляют и закрепляют теоретические знания. На этих занятиях студенты осваивают конкретные методы изучения дисциплины, обучаются экспериментальным способам анализа, умению работать с приборами и современным оборудованием. Лабораторные занятия дают наглядное представление об изучаемых явлениях и процессах, студенты осваивают постановку и ведение эксперимента, учатся умению наблюдать, оценивать полученные результаты, делать выводы и обобщения.
самостоя- тельная работа	Самостоятельная работа студентов по дидактической сути представляет собой комплекс условий обучения, организуемых преподавателем и направленных на самоподготовку учащихся. Учебная деятельность протекает без непосредственного участия преподавателя и заключается в проработке лекционного материала, подготовке к лабораторным занятиям; изучении учебной литературы из основного и дополнительного списка.
зачет	Зачет является формой оценки качества освоения студентом образовательной программы по дисциплине. По результатам зачета студенту выставляется оценка "зачтено" или "не зачтено". Зачет может проводиться в форме устного опроса по билетам (вопросам) или без билетов, с предварительной подготовкой или без подготовки, по усмотрению кафедры. Преподаватель может проставить зачет без опроса или собеседования тем студентам, которые активно участвовали на лабораторных занятиях.
экзамен	Экзамен по курсу проводится в виде тестирования или по билетам. При подготовке к экзамену необходимо опираться на источники, которые разбирались на лекциях в течение семестра. В каждом билете присутствует практическое задание (помимо 2 теоретических вопросов), таким образом, обучающийся демонстрирует и наработанные практические умения и навыки.

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 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 пол ки 6 шт. Шкаф двухстворчатый с полками 1 шт. Веб-камера 1 шт. Выход в Интернет, внутривузовская компьютерная сеть, доступ в электронную информационно-образовательную среду. Набор учебно-наглядных пособий: комплект презентаций в электронном формате по преподаваемой дисциплине 3-5 шт.

423600, Республика Татарстан, г. Елабуга, ул. Казанская, д. 89 ауд. 60

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 44.03.04 "Профессиональное обучение (по отраслям)" и профилю подготовки "Автоматизация энергетических систем".

Приложение №1
к рабочей программе дисциплины (модуля)
Б1.О.04.04 Программирование

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 (модулю)
Программирование

Направление подготовки: 44.03.04 - Профессиональное обучение (по отраслям)
Профиль подготовки: Автоматизация энергетических систем
Квалификация выпускника: бакалавр
Форма обучения: заочная
Язык обучения: русский
Год начала обучения по образовательной программе: 2026

СОДЕРЖАНИЕ

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

Код и наименование компетенции	Индикаторы достижения компетенций для данной дисциплины	Оценочные средства текущего контроля и промежуточной аттестации
УК-1 – - Способен осуществлять поиск, критический анализ и синтез информации, применять системный подход для решения поставленных задач	Знать принципы эффективного поиска, критического анализа и синтеза информации по основам программирования, методики системного подход к программированию, включая объектно-ориентированную, для решения задач на ЭВМ. Уметь осуществлять эффективный поиск, критический анализ и синтез информации по основам программирования, использовать методики системного подхода к программированию, включая объектно-ориентированную, для решения задач на ЭВМ. Владеть навыками эффективного поиска, критического анализа и синтеза информации по основам программирования; способностью применять методики системного подхода к программированию, включая объектно-ориентированную, для решения задач на ЭВМ.	**Текущий контроль:** Тестирование по темам Тема 1. Основы языка С++ 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 Лабораторные работы по темам Тема 1. Основы языка С++ 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 Контрольная работа по темам Тема 1. Основы языка С++ 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 **Промежуточная аттестация:** *Зачёт, экзамен*

2. Критерии оценивания сформированности компетенций

Компетенция	Зачтено			Не зачтено
	Высокий уровень (отлично)	**Средний уровень (хорошо)**	**Низкий уровень (удовлетворительно)**	**Ниже порогового уровня (неудовлетворительно)**
УК-1	Знает принципы эффективного поиска, критического анализа и синтеза информации по основам программирования, методики системного подхода к программированию, включая объектно-ориентированную, для решения задач на ЭВМ	Знает принципы поиска, критического анализа и синтеза информации по основам программирования, методики системного подход к программированию, программированию, включая объектно-ориентированную, допуская неточности в выборе методик для решения задач на ЭВМ.	Знает принципы поиска, критического анализа и синтеза информации по основам программирования, методики системный подход к программированию, допускает ошибки в знании основ языка программирования высокого уровня С++	Не знает принципы эффективного поиска, критического анализа и синтеза информации по основам программирования, применять методики системный подход к программированию, включая объектно-ориентированную, для решения задач на ЭВМ

	Умеет осуществлять эффективный поиск, критический анализ и синтез информации по основам программирования, использовать методики системный подход к программированию, включая объектно-ориентированную, для решения задач на ЭВМ.	Умеет осуществлять поиск, критический анализ и синтез информации по основам программирования, использовать методики системный подход к программированию, включая объектно-ориентированную, допуская неточности при решении задач на ЭВМ	Умеет осуществлять эффективный поиск, критический анализ и синтез информации по основам программирования, использовать методики системный подход к программированию, включая объектно-ориентированную, допуская ошибки при решении задач на ЭВМ.	Не осуществлять эффективный поиск, критический анализ и синтез информации по основам программирования, использовать методики системный подход к программированию, включая объектно-ориентированную, для решения задач на ЭВМ.
	Владеет навыками эффективного поиска, критического анализа и синтеза информации по основам программирования; способностью применять методики системного подхода к программированию, включая объектно-ориентированную, для решения задач на ЭВМ.	Владеет навыками эффективного поиска, критического анализа и синтеза информации по основам программирования; способностью применять методики системного подхода к программированию, включая объектно-ориентированную, допуская неточности при решении задач на ЭВМ	навыками эффективного поиска, критического анализа и синтеза информации по основам программирования; способностью применять методики системного подхода к программированию, включая объектно-ориентированную, допуская ошибки при решении задач на ЭВМ.	Не навыками эффективного поиска, критического анализа и синтеза информации по основам программирования; способностью применять методики системного подхода к программированию, включая объектно-ориентированную, для решения задач на ЭВМ.

3. Распределение оценок за формы текущего контроля и промежуточную аттестацию

3 курс (установочная и зимняя сессии) :
Текущий контроль:
Тестирование
Тема 1. Основы языка С++
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

Лабораторные работы
Тема 1. Основы языка С++
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

Контрольная работа
Тема 1. Основы языка С++
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

Выполнение каждого оценочного средства оценивается по шкале: отлично, хорошо, удовлетворительно, неудовлетворительно.
Общая оценка за текущий контроль представляет собой среднее значение между полученными оценками за все оценочные средства.

Промежуточная аттестация – зачёт, экзамен.
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 зачет (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.
Зачет (экзамен) проводится по билетам. В каждом билете два оценочных средства: устный или письменный ответ на вопрос и практическое задание.

Выполнение каждого задания за промежуточную аттестацию оценивается по шкале: отлично, хорошо, удовлетворительно, неудовлетворительно.
Общая оценка за промежуточную аттестацию представляет собой среднее значение между полученными оценками за все оценочные средства промежуточной аттестации.
В случае невозможности установления среднего значения оценки за промежуточную аттестацию (например, «хорошо» или «отлично»), итоговая оценка выставляется экзаменатором, исходя из принципа справедливости и беспристрастности на основании общего впечатления о качестве и добросовестности освоения обучающимся дисциплины (модуля).

Виды оценок:
Для зачета:
Зачтено
Не зачтено

Для экзамена:
Отлично
Хорошо
Удовлетворительно
Не удовлетворительно

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

4.1.1.1. Порядок проведения.
Тестирование проходит в письменной форме или с использованием компьютерных средств. Обучающийся получает определённое 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
Оценка «отлично» ставится, если обучающийся набрал:
86% правильных ответов и более.
Оценка «хорошо» ставится, если обучающийся набрал:
От 71% до 85 % правильных ответов.
Оценка «удовлетворительно» ставится, если обучающийся набрал:
От 56% до 70% правильных ответов.
Оценка «неудовлетворительно» ставится, если обучающийся набрал:
55% правильных ответов и менее.

4.1.1.3. Содержание оценочного средства

Темы 1-6

Формулировка задания

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 прозводного 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 С++
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

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

Оценка «отлично» ставится, если обучающийся:

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

Оценка «хорошо» 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.

Оценка «удовлетворительно» 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.

Оценка «неудовлетворительно» 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

Темы 1-6

Лабораторная работа 1. Основы языка С++.

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.
а) k=n+(++m);
б) m-->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 – длина окружности. Значение radius вводит пользователь.
Рассчитать площадь круга и длину окружности, записать в переменные c и s, 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 x1, x2.
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 «зациклить» – 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 betta, 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 alfa.
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) (большего из двух чисел k,m) оформить функцией.

2. Оформить функцию step(x,n) от вещественного x и целого n, вычисляющую (через последовательное умножение) xn 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 масивов 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 распознания 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,...,n найти все те, которые можно представить в виде суммы квадратов двух натуральных чисел. (Определить функцию, позволяющую распознавать полные квадраты.)
19. Даны действительные числа x1, y1, x2, y2,..., x10, y10. Найти периметр десятиугольника, вершины которого имеют соответственно координаты (x1, y1), (x2, y2),..., (x10, y10). (Определить функцию вычисления расстояния между двумя точками, заданными своими координатами.)
20. Дано натуральное число n. Выяснить, имеются ли среди чисел n, n+1,..., 2n близнецы, т.е. простые числа, разность между которыми равна двум. (Определить процедуру, позволяющую распознавать простые числа.)
В пяти следующих задачах потребуется 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.

Лабораторная работа 6. Классы и объекты в С++

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

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 gruppa[3];
gruppa[0].Set("Иванов",19,50);
```

и т.д.

или

```
STUDENT gruppa[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,
• cpp-файл с реализацией класса,
• cpp-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
стоимось – 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 работа7. 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), вынеся 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 С++ с использованием классов. Перегрузка операторов
Цель работы: Знакомство с объектно-ориентированным программированием в С++. Изучение общих понятий о классах - поля класса, методы класса (конструктор, деструктор и другие методы) [1, с. 264-289].
Изучение возможности перегрузки операторов в С++ [1, с. 294-303].
Задание. Согласно варианту (табл.16.1) задания написать программу на языке С++.
Таблица 16.1. Варианты исходных данных для лабораторной работы

Вариант 1

Класс	***Члены класса***	***Методы***	***Операторы перегрузки***
Комплексное число в алгебраической форме $a = x + y \cdot i$	Действи-тельная x и мнимая y части числа	Конструктор, деструктор, метод вычисления модуля комплексного числа и $\overline{a}$, 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 сокращение дроби, метод вывода дроби	Сложение (+), вычитание (-), умножение (*), деление (/) дробей
Исходные данные		***Результаты***	
Даны дроби $\frac{a}{b}, \frac{c}{d}, \frac{e}{f}, \frac{g}{h}, \frac{k}{l}$		$Z = \frac{\frac{a}{b} + \frac{c}{d}}{\frac{e}{f}} \cdot \left(\frac{g}{h} - \frac{k}{l}\right)$ На экран вывести несокращенную и сокращенную дробь Z	

Вариант 3			
Класс	***Члены класса***	***Методы***	***Операторы перегрузки***
Вектор	3 прямоугольные декартовые 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 $\lvert C\rvert$	

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^x прямых (| |), определение угла между 2-мя прямыми (%)
Исходные данные		***Результаты***	
Прямая AB: *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 gruz;} 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 gruz;

        };
```

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

```
#include "Lorry.h"
Lorry::Lorry(void):Car()
{
        gruz=0;
}
Lorry::~Lorry(void)
{
}
Lorry::Lorry(string M, int C, int P, int G):Car(M,C,P)
{
        gruz=G;
}
Lorry::Lorry(const Lorry &L)
{
        mark=L.mark; cyl=L.cyl;
        power=L.power;
        gruz=L.gruz;

}
void Lorry::Set_Gruz(int G)
{
        gruz=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.gruz; return in;
}
    ostream& operator<<(ostream&out,const Lorry&l)
{
        out<<"\nMARK : "<<l.mark;
        out<<"\nCYL : "<<l.cyl;
        out<<"\nPOWER : "<<l.power;
        out<<"\nGRUZ : "<<l.gruz; 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	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.
2	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.
3	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.

4	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)
5	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.
6	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.
7	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.
8	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.
9	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.

10	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.
11	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.
12	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.
13	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).
14	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.
15	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.

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?

Лабораторная работа 10. Шаблоны

1. ***Цель задания:***

1) Создание консольного приложения, состоящего из нескольких файлов в системе программирования Visual Studio.
2) Реализация шаблона класса-контейнера.

Ход работы

Задача

1. Класс- контейнер ВЕКТОР с элементами типа int. Реализовать операции:

[] – доступа по индексу;
() – определение размера вектора;
+ число – добавляет константу ко всем элементам вектора;

1. Пользовательский класс Time для работы с временными интервалами. Интервал должен быть представлен в виде двух полей: минуты типа int и секунды типа int.

при выводе минуты отделяются от секунд двоеточием.

1. Создать пустой проект.

2. Добавить в него файл Vector. h.
3. В файл Vector.h добавить описание параметризированного класса (шаблона)

Вектор:

```
#include <iostream>

using namespace std;

template <class T> //T - параметр шаблона

class Vector
{
public:
//конструктор с параметрами: выделяет память под s элементов и заполняет их //значением k
Vector(int s,T k);
//конструктор с параметрами
Vector(const Vector<T>&a);
//деструктор
~Vector();
//оператор присваивания
Vector&operator=(const Vector<T>&a); //операция доступа по индексу

T&operator[](int index);
//операция для добавление константы
Vector operator+(const T k);
//операция, возвращающая длину вектора
int operator()();
//перегруженные операции ввода-вывода
    //     <> - указывают на то, что функция является шаблоном
friend ostream& operator<< <>(ostream& out, const Vector<T>&a); friend istream& operator>>
<>(istream& in, Vector<T>&a); private:
```

```
int size;//размер вектора
T*data;//укзатель на динамический массив значений вектора
};
```

4. В этот же файл добавить определение методов параметризированного класса

Вектор:

```
/определение функций

//конструктор с параметрами

template <class T>
Vector<T>::Vector(int s,T k)
{
size=s;
data=new T[size];
for(int i=0;i<size;i++)
data[i]=k;
}
//конструктор копирования
template <class T>
Vector<T>::Vector(const Vector&a)
{
size=a.size;
data=new T[size];
for(int i=0;i<size;i++)
data[i]=a.data[i];
}
//деструктор
template <class T>
Vector<T>::~Vector()
{

delete[]data;
data=0;
}
//операция присваивания
template <class T>
Vector<T>&Vector<T>::operator=(const Vector<T>&a)
{
if(this==&a)return *this;
size=a.size;
if (data!=0) delete[]data;
data=new T[size];
for(int i=0;i<size;i++)
data[i]=a.data[i];
return *this;
}
//операция доступа по индексу
template <class T>
T&Vector<T>::operator[](int index)
{
if (index<size) return data[index];
else                    cout<<"\nError! Index>size";
}
//операция для добавления константы
template <class T>

Vector<T> Vector<T>::operator+(const T k)//+k {

Vector<T> temp(size,k);//инициализируем временный вектор любым значением for (int i=0;i<size;++i)
temp.data[i]=data[i]+k;
return temp;
}
```

```
//операция для получения длины вектора
template <class T>
int Vector<T>::operator ()()
{
return size;
}
//операции для ввода-вывода
template <class T>
ostream&operator<< (ostream&out,const Vector<T>&a)
{
for(int i=0;i<a.size;++i)
out<<a.data[i]<<" ";
return out;
}
template <class T>
istream&operator>> (istream&in,Vector<T>&a)
{
for(int i=0;i<a.size;++i)
in>>a.data[i];
return in;

}
```

5. Добавить в проект файл lab7_main().cpp с функцией main(), в которой выполнить тестирование параметризированного класса Вектор для

стандартного типа данных.

```
#include "Vector.h"

#include <iostream>
using namespace std;

void main()

{
//инициализация, ввод и вывод значений вектора

Vector<int>A(5,0);
cin>>A;
cout<<A<<endl;
//инициализация и вывод значений вектора Vector <int>B(10,1);
cout<<B<<endl;

//операция присваивания
B=A;
cout<<B<<endl;
//доступ по индексу
cout <<A[2]<<endl;
//получение длины вектора
cout<<"size="<<A()<<endl;
//операция сложения с константой
B=A+10;
cout<<B<<endl;
}
```

6. Добавить в проект класс Time с минимальной функциональностью (конструкторы, деструктор, операция присваивания, операции ввода-вывода).

Для этого в файл Time.h добавляем описание класса:

```
#include <iostream>

using namespace std;
class Time
{
public:
```

```
Time(void);
Time(int, int);
Time(const Time&);
Time&operator=(const Time&);
//перегруженные операции ввода-вывода

friend ostream& operator<< (ostream& out, const Time&); friend istream& operator>>
(istream& in, Time&); public:

virtual ~Time(void){};
private:
int min,sec;
};
```

В файл Time.cpp добавляем определение методов класса

```
#include "Time.h"

Time::Time(void)
{
min=sec=0;
}

Time::Time(int M, int S)

{
min=M;sec=S;
}
Time::Time(const Time&t)
{
min=t.min;
sec=t.sec;
}
Time&Time::operator =(const Time &t)
{
min=t.min;
sec=t.sec;
return*this;
}
ostream&operator<<(ostream&out, const Time&t)

{
out<<t.min<<" : "<<t.sec;

return out;
}
istream&operator>>(istream&in,Time&t)
{
cout<<"\nmin?"; in>>t.min;
cout<<"\nsec?";in>>t.sec;
return in;

}
```

7. В функцию main() добавить операторы для тестирования класса Time:

```
Time t;

cin>>t;

cout<<t;
```

8. В класс Time добавляем методы для реализации операций контейнера. В классе ВЕКТОР определена операция + число, которая добавляет константу ко всем элементам контейнера. Поэтому необходимо

определить операцию + число для класса Time. Это можно сделать следующим образом. В описание класса Time

(файл Time.h) добавить метод
Time operator+(Time k);

В файл Time.cpp добавить реализацию этого метода:
Time Time::operator+(Time k)

```
{
int t=min*60+sec;
int kt=k.min*60+k.sec;
t+=kt;
Time temp(t/60,t%60);
return temp;
}
```

9. Протестируем сложение для класса Time. Для этого фукцию main() добавим

операторы:

```
int k;

cout<<"k?";
cin>>k;
Time p;
p=t+k;

cout<<p;
```

10. Выполним тестирование параметризированного класса Вектор для

пользовательского типа данных Time.

```
. ......
Time t;

cin>>t;
cout<<t;

Vector<Time>A(5,t);

cin>>A;

cout<<A<<endl;

Vector <Time>B(10,t);

cout<<B<<endl;

B=A;

cout<<B<<endl;
cout <<A[2]<<endl;

cout<<"size="<<A()<<endl;

B=A+t;

cout<<B<<endl;
```

Варианты

№	Задание
1	Класс- контейнер ВЕКТОР с элементами типа int. Реализовать операции: [] – доступа по индексу;

() – определение размера вектора;

+ число – добавляет константу ко всем элементам вектора;

Пользовательский класс Time для работы с временными интервалами. Интервал должен быть представлен в виде двух полей: минуты типа int и секунды типа int. при выводе минуты отделяются от секунд двоеточием.

2 Класс- контейнер ВЕКТОР с элементами типа int.

Реализовать операции:

[]– доступа по индексу;

int() – определение размера вектора;

+ вектор – сложение элементов векторов a[i]+b[i];

Пользовательский класс Time для работы с временными интервалами. Интервал должен быть представлен в виде двух полей: минуты типа int и секунды типа int. при выводе минуты отделяются от секунд двоеточием.

3 Класс- контейнер ВЕКТОР с элементами типа int.

Реализовать операции:

[] – доступа по индексу;

+ вектор – сложение элементов векторов a[i]+b[i];

+ число – добавляет константу ко всем элементам вектора;

Пользовательский класс Time для работы с временными интервалами. Интервал должен быть представлен в виде двух полей: минуты типа int и секунды типа int. при выводе минуты отделяются от секунд двоеточием.

4 Класс- контейнер ВЕКТОР с элементами типа int.

Реализовать операции:

[] – доступа по индексу;

() – определение размера вектора;

* число – умножает все элементы вектора на число;

Пользовательский класс Time для работы с временными интервалами. Интервал должен быть представлен в виде двух полей: минуты типа int и секунды типа int.

при выводе минуты отделяются от секунд двоеточием.

5 Класс- контейнер ВЕКТОР с элементами типа int.

Реализовать операции:

[] – доступа по индексу;

int() – определение размера вектора;

* вектор – умножение элементов векторов a[i]*b[i];

Пользовательский класс Time для работы с временными интервалами. Интервал должен быть представлен в виде двух полей: минуты типа int и секунды типа int. при выводе минуты отделяются от секунд двоеточием.

6 Класс- контейнер МНОЖЕСТВО с элементами типа int.

Реализовать операции:

[] – доступа по индексу;

() – определение размера множества;

+ – объединение множеств;

Пользовательский класс Money для работы с денежными суммами. Число должно быть представлено двумя полями: типа long для рублей и типа int для копеек. Дробная часть числа при выводе на экран должна быть отделена от целой части запятой.

7 Класс- контейнер МНОЖЕСТВО с элементами типа int.

Реализовать операции:
[] – доступа по индексу;
int() – определение размера вектора;
* – пересечение множеств;

Пользовательский класс Money для работы с денежными суммами. Число должно быть представлено двумя полями: типа long для рублей и типа int для копеек. Дробная часть числа при выводе на экран должна быть отделена от целой части запятой.

8 Класс- контейнер МНОЖЕСТВО с элементами типа int.

Реализовать операции:
[] – доступа по индексу;
== - проверка на равенство;
> число – принадлежность числа множеству;

Пользовательский класс Money для работы с денежными суммами. Число должно быть представлено двумя полями: типа long для рублей и типа int для копеек. Дробная часть числа при выводе на экран должна быть отделена от целой части запятой.

9 Класс- контейнер МНОЖЕСТВО с элементами типа int.

Реализовать операции:
[] – доступа по индексу;
!= - проверка на неравенство;
< число – принадлежность числа множеству;

Пользовательский класс Money для работы с денежными суммами. Число должно быть представлено двумя полями: типа long для рублей и типа int для копеек. Дробная часть числа при выводе на экран должна быть отделена от целой части запятой.

10 Класс- контейнер МНОЖЕСТВО с элементами типа int. Реализовать операции:
[] – доступа по индексу;
() – определение размера вектора; - – разность множеств;

Пользовательский класс Money для работы с денежными суммами. Число должно быть представлено двумя полями: типа long для рублей и типа int для копеек. Дробная часть числа при выводе на экран должна быть отделена от целой части запятой.

11 Класс- контейнер СПИСОК с ключевыми значениями типа int. Реализовать операции:

[]– доступа по индексу;
int() – определение размера списка;
+ вектор – сложение элементов списков a[i]+b[i];

Пользовательский класс Money для работы с денежными суммами. Число должно быть представлено двумя полями: типа long для рублей и типа int для копеек. Дробная часть числа при выводе на экран должна быть отделена от целой части запятой.

12 Класс- контейнер СПИСОК с ключевыми значениями типа int. Реализовать операции:

[] – доступа по индексу;
() – определение размера вектора;

+ число – добавляет константу ко всем элементам вектора;

Пользовательский класс Pair (пара чисел). Пара должна быть представлено двумя полями: типа int для первого числа и типа double для второго. Первое число при выводе на экран должно быть отделено от второго числа двоеточием.

13 Класс- контейнер СПИСОК с ключевыми значениями типа int. Реализовать операции:
[] – доступа по индексу;
+ вектор – сложение элементов списков a[i]+b[i];
+ число – добавляет константу ко всем элементам списка;

Пользовательский класс Pair (пара чисел). Пара должна быть представлено двумя полями: типа int для первого числа и типа double для второго. Первое число при выводе на экран должно быть отделено от второго числа двоеточием.

14 Класс- контейнер СПИСОК с ключевыми значениями типа int. Реализовать операции:
[] – доступа по индексу;
() – определение размера списка;
* число – умножает все элементы списка на число;

Пользовательский класс Pair (пара чисел). Пара должна быть представлено двумя полями: типа int для первого числа и типа double для второго. Первое число при выводе на экран должно быть отделено от второго числа двоеточием.

15 Класс- контейнер СПИСОК с ключевыми значениями типа int. Реализовать операции:
[] – доступа по индексу;
int() – определение размера списка;
* вектор – умножение элементов списков a[i]*b[i];

Пользовательский класс Pair (пара чисел). Пара должна быть представлено двумя полями: типа int для первого числа и типа double для второго. Первое число при выводе на экран должно быть отделено от второго числа двоеточием.

Контрольные вопросы

1. В чем смысл использования шаблонов?
2. Каковы синтаксис/семантика шаблонов функций?
3. Каковы синтаксис/семантика шаблонов классов?
4. Что такое параметры шаблона функции?
5. Перечислите основные свойства параметров шаблона функции.
6. Как записывать параметр шаблона?
7. Можно ли перегружать параметризованные функции?
8. Перечислите основные свойства параметризованных классов.
9. Все ли компонентные функции параметризованного класса являются параметризованными?
10. Являются ли дружественные функции, описанные в параметризованном классе, параметризованными?
11. Могут ли шаблоны классов содержать виртуальные компонентные функции?
12. Как определяются компонентные функции параметризованных классов вне определения шаблона класса?
13. Что такое инстанцирование шаблона?
14. На каком этапе происходит генерирование определения класса по шаблону?

Содержание отчета

1) Постановка задачи (общая и конкретного варианта).
2) Описание параметризированного класса-контейнера.
3) Определение компонентных функций.
4) Описание пользовательского класса и его компонентных функций
5) Функция main().

6) Объяснение результатов работы программы.
7) Ответы на контрольные вопросы.

4.1.3. 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

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

Оценка «отлично» ставится, если обучающийся:
Правильно выполнил все задания. Продемонстрировал высокий уровень владения материалом. Проявлены превосходные способности применять знания и умения к выполнению конкретных заданий. .
Оценка «хорошо» 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.
Оценка «удовлетворительно» 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.
Оценка «неудовлетворительно» 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
Темы 1-6
Формулировка 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.

4.2. Оценочные средства промежуточной аттестации
По дисциплине предусмотрен зачёт и экзамен в 5 семестре. Зачёт (экзамен) проходит по билетам. В каждом билете один теоретический вопрос и одно практическое задание. Зачёт (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.
Устный или письменный ответ на вопрос направлен на проверку знаний основных разделов по дисциплине «Программирование».
4.2.1.2. Критерии оценивания.
Оценка «отлично» ставится, если обучающийся:
В ответе качественно раскрыл содержание темы. Ответ хорошо структурирован. Прекрасно освоен понятийный аппарат. Продемонстрирован высокий уровень понимания материала. Превосходное умение формулировать свои мысли, обсуждать дискуссионные положения.
Оценка «хорошо» ставится, если обучающийся:

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

Оценка «удовлетворительно» 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.

Оценка «неудовлетворительно» 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

Зачёт

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.

Экзамен

1. Объектно-ориентированное программирование.
2. Объявление класса.
3. Открытые и закрытые члены класса.
4. Статические члены класса.
5. Перегрузка методов.
6. Конструкторы. Создание и перегрузка конструктора.
7. Деструкторы. Правила создания деструктора. Использование деструкторов.
8. Внешняя операторная функция для переопределения бинарного оператора.
9. Перегрузка операторной функции.

10. Переопределение унарных операторов внешними функциями.
11. Перегрузка операторов методами класса.
12. Наследование классов и типы наследования.
13. Переопределение методов и виртуальные функции.
14. Многоуровневое наследование.
15. Чисто виртуальные методы и абстрактные классы.
16. Обобщённые функции. Перегрузка обобщённых функций.
17. Обобщённые классы. Шаблоны.

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.

Оценка «отлично» ставится, если обучающимся:

Задание выполнено полностью и правильно.

Оценка «хорошо» ставится, если обучающимся:

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

Оценка «удовлетворительно» ставится, если обучающимся:

Задание выполнено частично или с фактическими ошибками.

Оценка «неудовлетворительно» ставится, если обучающимся:

Задание не выполнено или выполнено с большим количеством фактических ошибок.

4.2.2.3. Оценочные средства.

Зачёт

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
$$1!+2!+3!+\ldots+n!,$$
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
$$\sqrt{(x1-x2)^2+(y1-y2)^2}$$
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

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

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:

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

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

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

Экзамен

1. 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.
2. 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.
3. 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.
4. 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.
5. 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.
6. 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.
7. 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.
8. 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.
9. Файл f1 состоит из целых чисел по 6 в строке. Переписать в файл f2 все положительные числа из файла f1, оставляя их в тех же строках.
   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.
10. Определить класс с именем Home, содержащий следующие поля: адрес дома; этажность; количество подъездов; количество квартир на этаже.

Реализовать методы:

- ввод данных в массив из n элементов типа Home;
- упорядочить по убыванию этажности;
- вывод информации об объекте, количество подъездов которого введено пользователем.

Приложение 2
к рабочей программе дисциплины (модуля)
Б1.О.04.04 Программирование

Перечень литературы, необходимой для освоения дисциплины (модуля)

Направление подготовки: 44.03.04 - Профессиональное обучение (по отраслям)

Профиль подготовки: Автоматизация энергетических систем

Квалификация выпускника: бакалавр

Форма обучения: заочная

Язык обучения: русский

Год начала обучения по образовательной программе: 2025

Основная литература:

1. Введение в основы программирования на C / Ю.А. Костиков, А.В. Мокряков, В.Ю. Павлов и др. - Москва : НИЦ ИНФРА-М, 2015. - 32 с. ISBN 978-5-16-103253-4. - Текст : электронный. - URL: https://znanium.com/read?id=6911 - Режим доступа: по подписке

2. Довек, Ж. Введение в теорию языков программирования / Довек Жиль, Леви Жан-Жак. — Москва : ДМК Пресс, 2013. — 134 с. - ISBN 978-5-94074-913-4. - Текст : электронный. - URL: https://znanium.com/read?id=34108 - Режим доступа: по подписке

3. Программирование. Процедурное программирование: Учебное пособие / Кучунова Е.В., Олейников Б.В., Чередниченко О.М. - Краснояр.:СФУ, 2016. - 92 с. - URL: http://znanium.com/bookread2.php?book=978627 -Режим доступа: по подписке

4. Царев, Р.Ю. Информатика и программирование [Электронный ресурс] : учеб. пособие / Р. Ю. Царев, А. Н. Пупков, В. В. Самарин, Е. В. Мыльникова. - Красноярск : Сиб. федер. ун-т, 2014. - 132 с. - ISBN 978-5-7638-3008-8. - Текст : электронный. - URL: https://znanium.com/read?id=126709 -Режим доступа: по подписке

Дополнительная литература:

1. Задачи по программированию / Под ред. Окулов С.М., - 3-е изд. - Москва :Лаборатория знаний, 2017. - 826 с.: ISBN 978-5-00101-448-5. - Текст : электронный. - URL: https://znanium.com/read?id=268319 -Режим доступа: по подписке

2. Кувшинов, Д. Р. Компьютерные науки : Основы программирования: Учебное пособие / Кувшинов Д.Р., - 2-е изд., стер. - Москва :Флинта, Изд-во Урал. ун-та, 2017. - 102 с. ISBN 978-5-9765-3144-4. - Текст : электронный. - URL: https://znanium.com/read?id=303926 -Режим доступа: по подписке

3. Подкур, М. Л. Программирование в среде Borland C++ Builder с математическими библиотеками MATLAB C/C++ [Электронный ресурс] / М. Л. Подкур, П. Н. Подкур, Н. К. Смоленцев. - М.: ДМК Пресс, 2009. - 496 с. - URL: http://znanium.com/bookread2.php?book=409183 -Режим доступа: по подписке

4. Программирование в алгоритмах / Окулов С.М., - 6-е изд., (эл.) - М.:Лаборатория знаний, 2017. - 386 с. - URL: http://znanium.com/bookread2.php?book=502153 -Режим доступа: по подписке

Приложение 3
к рабочей программе дисциплины (модуля)
Б1.О.04.04 Программирование

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

Направление подготовки: 44.03.04 - Профессиональное обучение (по отраслям)

Профиль подготовки: Автоматизация энергетических систем

Квалификация выпускника: бакалавр

Форма обучения: заочная

Язык обучения: русский

Год начала обучения по образовательной программе: 2025

Пакет офисного программного обеспечения Microsoft Office Professional plus 2010

MathCAD Education-University Edition

Notepad++ — свободный текстовый редактор с открытым исходным кодом для Windows

GIMP (GNU Image Manipulation Program («Гимп»)) — свободно распространяемый растровый графический редактор

Python – высокоуровневый язык программирования общего назначения

Inkscape (Инкскейп) — свободно распространяемый векторный графический редактор

Lazarus — открытая среда разработки программного обеспечения на языке Object Pascal для компилятора Free Pascal

Учебно-методическая литература для данной дисциплины имеется в наличии в электронно-библиотечной системе "ZNANIUM.COM", доступ к которой предоставлен обучающимся. ЭБС "ZNANIUM.COM" содержит произведения крупнейших российских учёных, руководителей государственных органов, преподавателей ведущих вузов страны, высококвалифицированных специалистов в различных сферах бизнеса. Фонд библиотеки сформирован с учетом всех изменений образовательных стандартов и включает учебники, учебные пособия, учебно-методические комплексы, монографии, авторефераты, диссертации, энциклопедии, словари и справочники, законодательно-нормативные документы, специальные периодические издания и издания, выпускаемые издательствами вузов. В настоящее время ЭБС ZNANIUM.COM соответствует всем требованиям федеральных государственных образовательных стандартов высшего образования (ФГОС ВО) нового поколения.