ПРИЛОЖЕНИЕ

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«РЯЗАНСКИЙ ГОСУДАРСТВЕННЫЙ РАДИОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»
ИМЕНИ. В.Ф. УТКИНА

Кафедра «Вычислительная и прикладная математика»

ОЦЕНОЧНЫЕ МАТЕРИАЛЫ ПО ДИСЦИПЛИНЕ
«Объектно-ориентированное программирование»

Направление подготовки
09.05.01 «Применение и эксплуатация автоматизированных систем специального назначения»

Направленность (профиль) подготовки
Математическое, программное и информационное обеспечение вычислительной техники и автоматизированных систем

Квалификация выпускника - специалист

Форма обучения - очная

Рязань

1. ОБЩИЕ ПОЛОЖЕНИЯ

Оценочные материалы - это совокупность учебно-методических материалов и процедур, предназначенных для оценки качестваосвоения обучающимися данной дисциплины как части основной образовательной программы.

Цель - оценить соответствие знаний, умений и уровня приобретенных компетенций, обучающихся целям и требованиям основной образовательной программы в ходе проведения текущего контроля и промежуточной аттестации.

Основная задача - обеспечить оценку уровня сформированности компетенций, приобретаемых обучающимся в соответствии с этими требованиями.

Контроль знаний обучающихся проводится в форме зачета с оценкой и курсовой работы в 3-м семестре.

2. ОПИСАНИЕ ПОКАЗАТЕЛЕЙ И КРИТЕРИЕВ ОЦЕНИВАНИЯ КОМПЕТЕНЦИЙ

Сформированность каждой компетенции в рамках освоения данной дисциплины оценивается по трехуровневой шкале:

1) пороговый уровень является обязательным для всех обучающихся по завершении освоения дисциплины;

2) продвинутый уровень характеризуется превышением минимальных характеристик сформированности компетенций по завершении освоения дисциплины;

3) эталонный уровень характеризуется максимально возможной выраженностью компетенций и является важным качественным ориентиром для самосовершенствования.

Уровень освоения компетенций, формируемых дисциплиной

а) описание критериев и шкалы оценивания тестирования:

Шкала оценивания	Критерий
3 балла (эталонный уровень)	уровень усвоения материала, предусмотренного программой: процент верных ответов на тестовые вопросы от 85 до 100%
2 балла (продвинутый уровень)	уровень усвоения материала, предусмотренного программой: процент верных ответов на тестовые вопросы от 75 до 84%
1 балл (пороговый уровень)	уровень усвоения материала, предусмотренного программой: процент верных ответов на тестовые вопросы от 60 до 74%
0 баллов	уровень усвоения материала, предусмотренного программой: процент верных ответов на тестовые вопросы от 0 до 59%

б) описание критериев и шкалы оценивания теоретического вопроса:

Шкала оценивания	Критерий
3 балла (эталонный уровень)	выставляется студенту, который дал полный ответ на вопрос, показал глубокие систематизированные знания, смог привести примеры, ответил на дополнительные вопросы преподавателя.
2 балла (продвинутый уровень)	выставляется студенту, который дал полный ответ на вопрос, но на некоторые дополнительные вопросы преподавателя ответил только с помощью наводящих вопросов.
1 балл (пороговый уровень)	выставляется студенту, который дал неполный ответ на вопрос в билете и смог ответить на дополнительные вопросы только с помощью преподавателя.
0 баллов	выставляется студенту, который не смог ответить на вопрос.

в) описание критериев и шкалы оценивания практического задания:

Шкала оценивания	Критерий
3 балла (эталонный уровень)	Задание решено верно.
2 балла (продвинутый уровень)	Задание решено верно, но имеются технические неточности в выполнении.
1 балл (пороговый уровень)	Задание решено верно, с дополнительными наводящими вопросами преподавателя.
0 баллов	Задание не решено.

На зачет с оценкой выносится: тестовое задание, 1 практическое задание и 1 теоретический вопрос. Студент может набрать максимум 9 баллов. Итоговый суммарный балл студента, полученный при прохождении промежуточной аттестации, переводится в традиционную форму по системе «отлично», «хорошо», «удовлетворительно», «неудовлетворительно». Обязательным условием получения оценок «отлично», «хорошо» и «удовлетворительно» является выполнениевсех предусмотренных в течение семестра практических заданий и лабораторных работ. Студент не выполнивший всех предусмотренных в течение семестра текущих заданий получает оценку «неудовлетворительно»

Шкала оценивания	Критерий
отлично (эталонный уровень)	8 - 9 баллов (все задания и лабораторные работы выполнены)
хорошо (продвинутый уровень)	6 - 7 баллов (все задания и лабораторные работы выполнены)
удовлетворительно (пороговый уровень)	4 - 5 баллов (все задания и лабораторные работы выполнены)
неудовлетворительно	0 - 3 баллов (студент не выполнил все задания и лабораторные работы)

Курсовая работа оценивается по принятой в ФГБОУ ВО «РГРТУ» четырехбалльной системе: «неудовлетворительно», «удовлетворительно», «хорошо» и «отлично».

Шкала оценивания	Критерий
«отлично»	*студент должен:* продемонстрировать глубокое усвоение материала; исчерпывающе, последовательно, грамотно и логически стройно изложить теоретический материал; правильно формулировать определения; уметь делать выводы по излагаемому материалу; безупречно ответить на дополнительные вопросы при защите курсовой работы в рамках рабочей программы дисциплины.
«хорошо»	*студент должен:* продемонстрировать достаточно полное знание материала; продемонстрировать знание основных теоретических понятий; достаточно последовательно, грамотно и логически стройно изложить материал; уметь сделать достаточно обоснованные выводы; ответить на все вопросы при защите курсовой работы; при этом возможны непринципиальные ошибки.
«удовлетворительно»	*студент должен:* продемонстрировать общее знание материала; знать основную рекомендуемую учебную литературу; уметь строить ответ в соответствии со структурой излагаемого вопроса; показать общее владение понятийным аппаратом дисциплины; уметь устранять допущенные ошибки в ответе на вопросы при

	защите курсовой работы;
«неудовлетворительно»	*ставится в случае:* незнания значительной части программного материала; не владения понятийным аппаратом; существенных ошибок при изложении учебного материала; неумения строить ответ в соответствии со структурой излагаемого вопроса; неумения делать выводы. Такая оценка ставится студентам, которые не могут продолжить обучение по данной образовательной программе, и если студент нарушил правила защиты курсовой работы (списывал и т.д.).

3. ПАСПОРТ ОЦЕНОЧНЫХ МАТЕРИАЛОВ ПО ДИСЦИПЛИНЕ

Контролируемые разделы (темы) дисциплины	**Код контролируемой компетенции (или её части)**	**Наименование оценочного средства**
Тема 1. Концепция ООП. Язык С++	ПК-4.1 ПК-4.2	Зачет с оценкой, КР
Тема 2. Пространства имен. Простые классы. Абстракция. Инкапсуляция.	ПК-4.1 ПК-4.2	Зачет с оценкой, КР
Тема 3. Перегрузка. Ссылки. Статические и константные члены класса.	ПК-4.1 ПК-4.2	Зачет с оценкой, КР
Тема 4. Наследование. Перегрузка методов. Дружественные функции и классы.	ПК-4.1 ПК-4.2	Зачет с оценкой, КР
Тема 5. Полиморфизм. Виртуальные функции. Абстрактные классы. Интерфейсы.	ПК-4.1 ПК-4.2	Зачет с оценкой, КР
Тема 6. Типы связей между объектами. Контейнерные классы.	ПК-4.1 ПК-4.2	Зачет с оценкой, КР
Тема 7. Множественное наследование. Шаблоны. Исключения.	ПК-4.1 ПК-4.2	Зачет с оценкой, КР
Тема 8. STL - библиотека шаблонов. Строковые классы.	ПК-4.1 ПК-4.2	Зачет с оценкой, КР

4. ТИПОВЫЕ КОНТРОЛЬНЫЕ ЗАДАНИЯ ИЛИ ИНЫЕ МАТЕРИАЛЫ

4.1 Промежуточная аттестация

Типовые тестовые вопросы закрытого типа:

ПК-4 Способен руководить и участвовать в процессе разработки программного обеспечения автоматизированной системы.
ПК-4.1 Выполняет формализацию и алгоритмизацию поставленных задач.

Вопрос 1. Что означает термин "алгоритмизация" в программировании?

Проектирование базы данных

Создание пользовательского интерфейса

Оптимизация производительности

Разработка последовательности действий, приводящих к решению задачи

Вопрос 2. Какой инструмент используется для алгоритмизации задач?

Интерпретатор
Блок-схема
Процессор
Компилятор

Вопрос 3. Что такое "псевдокод"?

Машинный код, выполняемый процессором
Спецификация программного обеспечения
Язык программирования
Формальное описание алгоритма на естественном языке

Вопрос 4. Что означает термин "формализация" в программировании?

Тестирование программы
Разработка пользовательского интерфейса
Определение требований к программе
Преобразование алгоритма в код

Вопрос 5. Что такое "структура данных"?

Организация и хранение данных
Файловая система
Аппаратный способ хранения данных
Программный интерфейс

Вопрос 6. Что такое "отладка"?

Формализация требований
Процесс исправления ошибок в программе
Разработка алгоритма программы
Создание пользовательского интерфейса

Вопрос 7. Какой из нижеперечисленных шагов является частью формализации задачи?

Разработка диаграммы классов
Написание комментариев в коде
Определение входных и выходных данных
Выбор языка программирования

Вопрос 8. Какие операторы нельзя перегружать?

>> :: . sizeof
?: :: * sizeof
?: :: . sizeof
?: ++ . sizeof

Вопрос 9. Уровень доступа к полям класса задаётся с помощью этих уровней доступа:

individual, covered, public
private, protected, public
sealed, readonly, anywhere
personal, defended, common

Вопрос 10. Статические поля класса позволяют:

Задать поля класса, доступные только из константных методов
Задать поля класса со статическим управлением памятью
Задать неизменяемые поля класса
Задать поля класса, которые доступны из всех экземпляров

Вопрос 11. Какие три основных принципа ООП?

переиспользование, функциональность, декларативность
инкапсуляция, наследование, полиморфизм
когерентность, абстрагирование, шаблонность
абстракция, агрегация, ассоциация

Вопрос 12. Деструктор класса объявляется как:

~ClassName(ClassName* this);
destructor ClassName(ClassName* this);
$ClassName();
~ClassName();

Вопрос 13. Стандартными потоками вводавывода в C++ являются:

cin и cout
puts и gets
iostream и stdio
printf и scanf

Вопрос 14. Пространства имен namespace позволяют:

объявлять новые члены класса
подключать новые библиотеки
указать компилятору какие имена нужно исключить на этапе компиляции
избежать конфликта имен

Вопрос 15. Зачем нужен конструктор копии:

позволяет включить для объекта автоматическую сборку мусора
если его не задать, объект нельзя будет скопировать
позволяет задать свой алгоритм копирования объекта, если по-умолчанию не подходит
позволяет сделать ссылку, два объекта будут ссылаться на одни и те же данные

Вопрос 16. nullptr это:

предотвращает запись в указатель, позволяет только читать данные
нужен для предотвращения приведения типа нулевого указателя к числу 0
присвоение к нему удаляет объект
псевдоним значения NULL

Вопрос 17. Класс, производящий наследование в потомках, называют:

надкласс
подкласс
абстрактный класс
производный класс

ПК-4 Способен руководить и участвовать в процессе разработки программного обеспечения автоматизированной системы.
ПК-4.2 Использует современные инструментальные средства разработки и языки программирования.

Вопрос 1. Какой язык программирования наиболее часто используется для анализа данных и машинного обучения?

Python
Java
Ruby
C#

Вопрос 2. Какой язык программирования используется для разработки системных утилит?

C/C++
Javascript
Kotlin
Python

Вопрос 3. Какая операционная система используется разработчиками мобильных приложений для iOS?

Windows
macOS
Linux
Android

Вопрос 4. Какой язык программирования используется для разработки веб-приложений?

Pascal
Swift
JavaScript
C++

Вопрос 5. Какая инструментальная среда разработки часто используется для создания баз данных?

PyCharm
Microsoft SQL Server Management Studio
Visual Studio Code
Node.js

Вопрос 6. Какой класс называют полиморфным?

класс, созданный с помощью оператора new
класс, содержащий хотя бы одну виртуальную функцию
класс, не имеющий дочерних классов
класс, не имеющий полей с данными

Вопрос 7. Контейнерные классы подразделяются на:

контейнеры-переменные и контейнеры-константы
контейнеры-ссылки и контейнеры-указатели
контейнеры-значения и контейнеры-ссылки
контейнеры-значения и контейнеры-литералы

Вопрос 8. Специальный (ad-hoc) полиморфизм в C++ реализуется в виде:

наследования

полиморфизма подтипов

шаблонов

параметры методов

Вопрос 9. Ключевое слово override в методах позволяет:

перегружать методы класса

отказаться от виртуального переопределения

объявлять методы, тела которых объявлены в дочерних классах

подтвердить совпадение сигнатур методов, и, следовательно, избежать ошибок, связанных с виртуальным переопределением метода

Вопрос 10. Интерфейсный класс это:

класс, предоставляющий доступ к графическому выводу на экран компьютера

класс, который не имеет полей (переменных-членов), и все методы которого являются чисто виртуальными

класс, содержащий только статические поля и методы

класс, реализующий графический интерфейс пользователя

Вопрос 11. Тип связи включение подразделяется на:

композицию и агрегацию

агрегацию и зависимость

композицию и ассоциацию

наследование и добавление

Вопрос 12. Истинный (параметрический) полиморфизм в C++ реализуется в виде:

наследования

параметры методов

шаблонов

полиморфизма подтипов

Вопрос 13. Ключевое слово auto нужно для:

автоматического вычисления полей-геттеров класса

автоматического управления памятью с помощью сборщика мусора

автоматического вывода типа компилятором

автоматического выделения памяти

Вопрос 14. Какие бывают типы связей между объектами классов:

включение, ассоциация, зависимость, наследование

зависимость, тождественность, ссылка, наследование

добавление, композиция, включение, ссылка

перегрузка, полиморфизм, компонентность, агрегация

Вопрос 15. Что такое виртуальная функция?

функция встраиваемая по месту вызова

при вызове выполняет наиболее «дочернюю» реализацию

функция не имеющая тела в данном классе

функция наследуемая от другого класса

Вопрос 16. Как правильно удалить массив array[] с помощью оператора delete:

delete [] array;
delete array[];
delete array;
delete &array;

Типовые тестовые вопросы открытого типа:

ПК-4 Способен руководить и участвовать в процессе разработки программного обеспечения автоматизированной системы.
ПК-4.1 Выполняет формализацию и алгоритмизацию поставленных задач.

Вопрос 1. Какие критерии помогают программисту определить, что задача успешно сформализована?
Ответ: Задача считается успешно сформализованной, если она ясно и точно определена, все требования учтены, алгоритмы разработаны и структуры данных выбраны соответствующим образом. Критерии успешной формализации включают полноту, однозначность, адекватность и проверяемость задачи.

Вопрос 2. Какие проблемы могут возникнуть при формализации сложных задач в программировании и как их можно решить?
Ответ: При формализации сложных задач могут возникнуть проблемы с определением требований, разработкой эффективных алгоритмов, выбором подходящих структур данных и обработкой больших объемов данных. Эти проблемы могут быть решены путем тщательного анализа задачи, использования алгоритмических методов и структур данных, а также оптимизации процесса обработки данных.

Вопрос 3. Какие инструменты и методы используются программистами для формализации задач?
Ответ: Программисты используют различные инструменты и методы для формализации задач, такие как диаграммы потоков данных, диаграммы классов, диаграммы последовательности, языки моделирования (например, UML), структурированные текстовые описания и спецификации требований.

Вопрос 4. Какие преимущества предоставляет алгоритмизация задач в программировании?
Ответ: Алгоритмизация задач в программировании позволяет программистам разбить сложные задачи на более простые и понятные шаги. Это упрощает процесс разработки, повышает читаемость кода, облегчает отладку и тестирование программы, а также способствует повторному использованию кода.

Вопрос 5. Какие роли играют диаграммы и моделирование в процессе формализации и алгоритмизации задач?
Ответ: Диаграммы и моделирование играют важную роль в процессе формализации и алгоритмизации задач. Они помогают визуализировать структуру и взаимодействие компонентов системы, определить последовательность действий, выделить ключевые аспекты задачи и облегчить коммуникацию между разработчиками и заказчиками.

Вопрос 6. Какие шаги включает процесс формализации задачи перед ее реализацией?
Ответ: Процесс формализации задачи включает следующие шаги: определение требований,

анализ и моделирование задачи, разработку алгоритма, выбор структур данных, определение входных и выходных данных, а также документирование и проверку формализованной задачи.

Вопрос 7. Какие навыки и знания необходимы программисту для успешной формализации и алгоритмизации задач?
Ответ: Для успешной формализации и алгоритмизации задач программисту необходимы навыки анализа требований, знание языков моделирования и спецификаций, понимание алгоритмических методов и структур данных, а также опыт работы с инструментами разработки и моделирования. Также важны коммуникативные навыки для взаимодействия с заказчиками и другими разработчиками.

Вопрос 8. Как формализация и алгоритмизация задач влияют на процесс разработки программного обеспечения?
Ответ: Формализация и алгоритмизация задач значительно упрощают процесс разработки программного обеспечения. Они помогают программистам лучше понять требования и цели проекта, определить структуру программы, разработать эффективные алгоритмы и выбрать подходящие структуры данных. Это ускоряет разработку, повышает качество программы и уменьшает количество ошибок.

Вопрос 9. Что подразумевается под формализацией задач в программировании и почему она важна?
Ответ: Формализация задач в программировании означает преобразование неструктурированных или нечетких требований в явные и точные спецификации. Это важно, потому что формализация позволяет программистам лучше понять задачу, определить ее границы и требования, а также уменьшить возможность ошибок и неоднозначностей.

Вопрос 10. Какие методы тестирования могут быть применены для проверки правильности формализации и алгоритмизации задач?
Ответ: Для проверки правильности формализации и алгоритмизации задач могут быть применены различные методы тестирования, такие как модульное тестирование, интеграционное тестирование, системное тестирование и приемочное тестирование. Тестирование должно включать проверку соответствия результатов выполнения программы ожидаемым результатам и соответствие требованиям.

Вопрос 11. Что такое наследование?
Ответ: Наследование — это механизм создания нового класса на основе уже существующего. К существующему классу могут быть добавлены новые поля и методы, либо расширены уже существующие методы. Позволяет повторно использовать функционал уже существующих классов, когда создаваемый класс является вариантом уже существующего.

Вопрос 12. Для чего нужен this?
Ответ: В любой метод класса (включая конструкторы и деструкторы) передаётся неявный указатель this указывающий на сам объект данного класса. С его помощью метод класса определяет, с каким объектом ему предстоит работать.

Вопрос 13. Что такое ООП?
Ответ: Объектно-ориентированное программирование (ООП) — методология программирования, основанная на представлении программы в виде совокупности взаимодействующих друг с другом объектов, каждый из которых является экземпляром

определённого класса, которые, образуют свою иерархию наследования

Вопрос 14. Для чего нужны операторы new и delete?
Ответ: C++, в отличии от C, содержит новые два оператора - new и delete, выполняющих работу с динамической памятью более эффективно и просто. Преимущества newdelete: Автоматическое вычисление размера необходимой памяти, Не надо использовать преобразование типа указателя, new не только выделяет память, но и вызывает конструкторыдеструкторы объектов, Можно перегрузить

Вопрос 15. Что такое деструктор?
Ответ: Деструктор – это специальный метод, который вызывается автоматически при удалении экземпляра класса. Часто используется для освобождения памяти, динамически выделенной в конструкторе класса.

Вопрос 16. В чем отличие структур и классов в C++?
Ответ: В C++ структуры (struct) аналогичны классам, только по умолчанию у них все поля public, а не private.

Вопрос 17. Что такое перегрузка методов?
Ответ: определение нескольких методов с одинаковым именем, но различными параметрами. Список параметров может отличаться порядком следования, количеством, типом. Перегрузка нужна для того, чтобы избежать дублирования имён методов, выполняющих сходные действия, но с различной реализацией.

Вопрос 18. Чем отличается объект от класса?
Ответ: Объект — это некоторая сущность, обладающая определённым состоянием и поведением, имеющая определённые свойства (поля) и операции над ними (методы). Является экземпляром своего класса. Класс — представляет собой некоторый общий «шаблон» для создания объектов, инициализацию полей и описания поведения – своих методов. Фактически класс является типом для объекта (экземпляра класса).

Вопрос 19. Что такое дружественный класс?
Ответ: Дружественный класс – класс, имеющий доступ к приватным полям другого класса.

Вопрос 20. Для чего нужно разделять класс на реализацию cpp и заголовочный файл h?
Ответ: классы в C++ обычно разделяются по файлам .h и .cpp (соответственно интерфейсную часть и реализацию).В интерфейсной части пишется описание класса, его полей и заголовков методов, в реализации – тела методов. Это позволяет осуществлять раздельную компиляцию отдельных модулей и не перекомпилировать каждый раз весь проект заново).

Вопрос 21. Чем характеризуются константные объекты класса?
Ответ: Объекты классов можно объявить константными. Их инициализация выполняется через конструкторы, после этого любая попытка изменить переменные-члены объекта запрещена (как напрямую, так и через методы). Константные объекты класса могут вызывать только константные методы класса - эти методы гарантируют, что не будут изменять сам объект или вызывать другие неконстантные методы

Вопрос 22. Что такое ссылки?
Ответ: Ссылка — это тип переменной в языке C++, который работает как псевдоним другого

объекта или значения, объявляются через &. Должны быть инициализированы при создании, и не могут быть изменены после. В отличие от указателей, не могут быть нулевыми. Для доступа к значению по ссылке не требуется операция разыменовывания, т.к. это просто псевдоним переменной.

Вопрос 23. Что такое конструктор?
Ответ: Конструктор - это специальный метод, который вызывается автоматически при создании экземпляра класса. При этом память под него уже выделена заранее. Позволяет выполнить сложный код для инициализации полей. Если в классе не определен явно – будет сгенерирован по-умолчанию.

ПК-4 Способен руководить и участвовать в процессе разработки программного обеспечения автоматизированной системы.
ПК-4.2 Использует современные инструментальные средства разработки и языки программирования.

Вопрос 1. Какие инструменты используют программисты для тестирования кода и какие возможности они предоставляют?
Ответ: Программисты используют различные инструменты для тестирования кода, такие как фреймворки для модульного тестирования (например, JUnit для Java), инструменты для автоматизированного тестирования интерфейсов (например, Selenium) и инструменты для тестирования производительности (например, Apache JMeter). Эти инструменты позволяют программистам проверять правильность работы кода, обнаруживать ошибки и улучшать качество программного обеспечения.

Вопрос 2. Какие современные инструментальные средства разработки используют программисты и какие преимущества они предоставляют?
Ответ: Программисты используют различные современные инструментальные средства разработки, такие как интегрированные среды разработки (IDE), системы контроля версий (VCS), отладчики и средства автоматизации сборки. Эти инструменты облегчают процесс разработки, повышают производительность, обеспечивают удобную навигацию по коду и помогают в обнаружении и исправлении ошибок.

Вопрос 3. Какие инструменты используют программисты для управления зависимостями и сборки проектов?
Ответ: Программисты используют инструменты для управления зависимостями и сборки проектов, такие как Apache Maven, Gradle и npm. Эти инструменты позволяют управлять зависимостями проекта, загружать и устанавливать необходимые библиотеки и модули, а также автоматизировать процесс сборки и развертывания приложений.

Вопрос 4. Какие инструменты используют программисты для статического анализа кода и какие преимущества они предоставляют?
Ответ: Программисты используют инструменты для статического анализа кода, такие как SonarQube, FindBugs и ESLint, для обнаружения потенциальных проблем в коде, таких как ошибки, уязвимости и нарушения стандартов кодирования. Эти инструменты помогают повысить качество кода, улучшить его читаемость и обнаружить потенциальные проблемы без необходимости выполнения кода.

Вопрос 5. Какие преимущества предоставляет использование интегрированных сред разработки (IDE)?

Ответ: Интегрированные среды разработки (IDE) предоставляют программистам множество преимуществ. Они обеспечивают удобную среду для написания, отладки и тестирования кода, предлагают автодополнение, подсветку синтаксиса и другие инструменты для повышения производительности и качества кода. IDE также облегчают работу с системами контроля версий и интеграцию с другими инструментами разработки.

Вопрос 6. Какие языки программирования считаются современными и широко используются программистами?
Ответ: Среди современных языков программирования, широко используемых программистами, можно выделить Python, JavaScript, Java, C++, C#, Ruby, Go, Swift и Kotlin. Эти языки обладают различными особенностями и предназначены для разработки различных типов приложений.

Вопрос 7. Какие инструменты используют программисты для разработки пользовательского интерфейса и какие возможности они предоставляют?
Ответ: Программисты используют различные инструменты для разработки пользовательского интерфейса, такие как фреймворки для веб-разработки (например, React, Angular, Vue.js), инструменты для создания мобильных приложений (например, Flutter, React Native) и инструменты для разработки настольных приложений (например, JavaFX, Electron). Эти инструменты предоставляют возможности для создания интерактивных и привлекательных пользовательских интерфейсов, упрощают процесс разработки и обеспечивают переносимость приложений на различные платформы.

Вопрос 8. Какие средства автоматизации сборки используют программисты и какие преимущества они предоставляют?
Ответ: Программисты используют средства автоматизации сборки, такие как Apache Maven, Gradle и Make, для автоматизации процесса сборки программного кода. Эти инструменты позволяют управлять зависимостями, компилировать исходный код, создавать исполняемые файлы и упрощают процесс развертывания приложений. Автоматизация сборки ускоряет процесс разработки и обеспечивает консистентность и надежность сборки.

Вопрос 9. Какие системы контроля версий (VCS) широко используются программистами и какие преимущества они предоставляют?
Ответ: Системы контроля версий (VCS), такие как Git, Subversion (SVN) и Mercurial, широко используются программистами. Они позволяют отслеживать изменения в коде, управлять версиями проекта, совместно работать над кодом и упрощают процесс слияния изменений. VCS также обеспечивают резервное копирование кода и восстановление предыдущих версий.

Вопрос 10. Какие инструменты используют программисты для отладки кода и какие возможности они предоставляют?
Ответ: Программисты используют различные инструменты для отладки кода, такие как отладчики, логгеры и профилировщики. Отладчики позволяют программистам исследовать и исправлять ошибки в коде, устанавливать точки останова и следить за значением переменных. Логгеры помогают отслеживать выполнение программы и записывать сообщения об ошибках или событиях. Профилировщики позволяют анализировать производительность программы и оптимизировать ее работу.

Вопрос 11. Зачем нужно ключевое слово final в объявлении классов?
Ответ: final используется в объявлении класса после его имени для запрещения наследования

Вопрос 12. Что такое полиморфизм?
Ответ: Полиморфизм – это семейство механизмов языка программирования, позволяющее работать одному коду с различными типами, либо иметь одинаковый интерфейс доступа для различных реализаций.

Вопрос 13. Приведите пример объявления копирующего конструктора
Ответ:

```
ClassName(const ClassName& other)
{
}
```

Вопрос 14. Что такое тип связи композиция?
Ответ: Композиция (включение по значению) - включаемый объект может существовать только как часть контейнера. Если контейнер будет уничтожен, то и включённый объект тоже будет уничтожен. Объект ничего не знает о контейнере.

Вопрос 15. Что такое тип связи агрегация?
Ответ: Агрегация (включение по ссылке) - отношение между двумя равноправными объектами, когда объект-контейнер имеет только ссылку на другой объект. Оба объекта могут существовать независимо: если контейнер будет уничтожен, то его содержимое — нет. Но объекты по прежнему не знают о существовании контейнера.

Вопрос 16. Что такое чисто виртуальная функция?
Ответ: C++ позволяет создавать т.н. чистые виртуальные функции (или «абстрактные функции»), которые не имеют определения в родительском классе, и их обязательно переопределяют дочерние классы. Для её объявления достаточно заголовок приравнять к нулю

Вопрос 17. Что такое абстрактный класс?
Ответ: Абстрактный класс – класс, содержащий хотя бы одну чистую виртуальную функцию. Экземпляры абстрактного класса создавать нельзя.

Вопрос 18. Приведите пример объявления класса со статическими полями и методами:
Ответ:

```
class Object
{
int field1;
float field2;
public:
void method()
        {
                //....
        }
};
```

Вопрос 19. Зачем нужен виртуальный деструктор?
Ответ: Деструктор полиморфного базового класса должен всегда объявляться виртуальным. Только так обеспечивается корректное разрушение объекта производного класса через указатель на соответствующий базовый класс.

Вопрос 20. Приведите пример создания и удаления экземпляра класса в динамической памяти в C++
Ответ:
```
Class* object = new Class();
// ...
delete object;
object = nullptr;
```

Вопрос 21. Что такое полиморфизм подтипов?
Ответ: Полиморфизм подтипов означает, что если функция принимает некий базовый тип, то она должна также работать и со всеми его подтипами, её поведение сохранит логику и идентичность. Действительный тип объекта может при этом быть скрыт как «чёрный ящик», и предоставляться лишь по запросу идентификации объекта.

Вопрос 22. Приведите пример объявления шаблона класса
Ответ:
```
template <typename T>
class MyClass
{
* elems;
};
```

Вопрос 23. Приведите пример перегрузки оператора [] для класса
Ответ:
```
class Class
{
public:
operator[] (int i) const
        {
                // ....
        }
};
```

4.2 Типовые контрольные вопросы и задания

1. Что такое контейнерные классы и какие виды их бывают?
2. Что такое конструктор копирования? В каких случаях он объявляется явно?
3. Что такое идиома RAII?
4. Что такое адаптеры? Какие есть стандартные адаптеры в STL?
5. Что такое тип связи ассоциация? Приведите пример.36. Что такое тип связи зависимость? Приведите пример.
6. Что такое тип связи композиция? Приведите пример.
7. Что такое чистые виртуальные функции? Что такое абстрактный класс и для чего он используется?
8. Что такое перегрузка функций? Как компилятор определяет какая функция вызывается? Зачем был введен nullptr и в чём его отличие от NULL?

9. Что такое non-type параметры шаблона, какого типа они могут быть? Что такое явная специализация шаблона и каким образом она объявляется?
10. Что такое циклическая зависимость, к каким проблемам она может приводить и какие пути решения существуют?
11. Какие перегруженные операторы имеют итераторы?
12. Что такое алгоритмы STL и как они реализованы?
13. Зачем нужна интерфейсная часть и часть реализации при объявлении класса в программе?
14. Что такое множественное наследование? Что такое виртуальное наследование и для чего оно применяется в языке C++?
15. Что такое инстанцирование шаблона и в какой момент оно происходит? Какие преимущества и недостатки есть у шаблонов?
16. Какие рекомендации существуют по объявлению классов?
17. Что такое исключения? Какова общая структура выбрасывания и обработки исключения? Из каких блоков она состоит?
18. Что такое объект и что такое класс? Дайте определение принципа ООП абстракции и инкапсуляции.
19. Для чего используется виртуальный деструктор? Для чего нужны модификаторы override и final?
20. Что такое виртуальная функция? В чём отличие виртуальных функций от обычного переопределения?
21. Что такое полиморфизм? Какие виды полиморфизма бывают?
22. Что такое виртуальная функция? Как добавить определение по умолчанию для виртуальной функции?
23. С помощью каких операторов происходит выделениеосвобождение динамической памяти в C++? В чём их преимущество перед функциями из C?
24. Что такое перегрузка операторов и какие её виды бывают? Как она объявляется? Какие операторы нельзя перегружать?
25. Что такое enum class и в чём его отличие от обычного enum?
26. Что такое полиморфизм? Что такое полиморфизм подтипов? Какой класс называют полиморфным?
27. Какие есть основные методы для работы с итераторами?
28. Что такое потоки ввода-вывода и как с ними работать в C++?
29. Что такое статические поля и методы класса? Зачем они используются?
30. Зачем используются ссылки как параметры функции? А в качестве возвращаемого значения?
31. Чем характеризуется приведение типов с помощью static_cast и dynamic_cast?
32. Что такое исключения? Для чего нужны классы-исключений? Каковы недостатки исключений?
33. Что такое STL и из каких частей она состоит? Что такое ассоциативные контейнеры?
34. Что такое лямбда-выражения и какой формат их объявления?
35. Что такое частичная специализация шаблона и каким образом она объявляется? Какое ограничение существует у частичной специализации шаблона и каким образом его можно обходить?
36. Что такое переопределение метода и зачем оно используется? Зачем нужно ключевое слово auto?
37. Для чего нужны пространства имен и как их объявлять? Для чего нужны using-объявления и как их использовать?
38. Что такое наследование и для чего оно нужно? Что такое суперкласс? Подкласс? Базовый класс?

39. Что такое шаблоны и для чего они применяются? Как объявить шаблон функции и шаблон класса?
40. Какие типы итераторов обычно предоставляют контейнеры?
41. Что такое интерфейсный класс и для чего он используется?
42. Какие рекомендуемые умные указатели есть в стандартной библиотеке?
43. Что такое тип связи агрегация? Приведите пример.
44. Что такое ссылки и в чём их отличие от указателей? В каких случаях надо применять ссылки, а в каких указатели?
45. Что такое исключения? В каких случаях стоит использовать исключения?
46. Что такое указатель this и в каких случаях он применяется?
47. Что такое юнит-тестирование? На какие части обычно разделен код теста?
48. Дайте краткие сведения о языке C++. Чем он отличается от C?
49. Что такое тип связи наследование? Приведите пример.
50. Что такое константные методы класса и для чего они нужны?
51. Что такое множественное наследование? Что такое ромбовидное наследование и какие проблемы с этим связаны?
52. Что такое конструктор и деструктор? Формат их объявления.
53. Что такое умный указатель и для чего он используется?
54. Чем характеризуется приведение типов с помощью const_cast и reinterpret_cast?
55. Что такое STL и из каких частей она состоит? Какие есть стандартные последовательные контейнеры в STL?
56. Какие достоинства и недостатки ООП вы знаете?
57. Что такое исключения? В чем преимущество исключений перед возвратом кодов ошибок?
58. Что такое дружественная функция и класс? Когда их стоит использовать?
59. Что такое STL и из каких частей она состоит? Какие есть стандартные ассоциативные контейнеры в STL?
60. Что такое объект std::initializer_list и для чего он используется?
61. Какие уровни доступа есть при объявлении класса? Формат объявления класса.
62. Что такое объект std::string и какие основные методы имеет?
63. Что такое объектно-ориентированное программирование? Какова цель внедрения ООП?
64. Что такое STL и из каких частей она состоит? Что такое последовательные контейнеры?
65. Что такое объект и что такое класс? Дайте определение принципа ООП наследования и полиморфизма.
66. Что такое итераторы и для чего они применяются?
67. В каком порядке при наследовании вызываются конструкторы и деструкторы? Какие типы наследования бывают?

4.3 Типовые задачи по дисциплине

1. «Булева матрица» – BoolMatrix. Разработать класс представляющий собой матрицу булевых значений размерности nxm. Реализовать функции для изменения и получения значения указанного элемента, логического сложения, умножения и инверсии матриц. Реализовать функцию для подсчета количества true и false значений в матрице. Предусмотреть функцию toString. Создать 2 массива классов и с помощью них поэлементно показать работу всех операций.

2. «Множество целых чисел» –Set. Разработать класс множества целых чисел мощности n. Реализовать функции для определения принадлежности заданного элемента множеству, добавлениедаление элемента, пересечения, объединения, разности двух множеств.

Предусмотреть функцию toString. Создать 2 массива классов и с помощью них поэлементно показать работу всех операций.

3. «Массив строк» – StringArray. Разработать класс для представления массива строк. Реализовать функции для добавлениядаления строк, для поэлементной конкатенации двух массивов, упорядочения строк по длине, слияния двух массивов строк с удалением повторяющихся строк, а также формирование массива количества слов в каждой строке. Предусмотреть функцию toString. Создать 2 класса и с помощью них поэлементно показать работу всех функций.

4. «Вектор» – Vector. Разработать класс вектора размерности n. Реализовать функции для изменения и получения значения компонента вектора, вычисления длины вектора, скалярного произведения, сложения, умножения, умножения на скаляр. Предусмотреть функцию toString. Создать 2 массива классов и с помощью них поэлементно показать работу всех операций.

5. Описать массив классов и поместить в него сгенерированные сведения о N спортсменах. Предусмотреть такие сведения как ФИО спортсмена (отдельным объектом), дата рождения (отдельный объект), страна, вид соревнования, результаты соревнований (отдельный объект со сведениями о названии соревнования, дате проведения, результате спортсмена). Написать функцию выдачи списка спортсменов по названию соревнования, стране или диапазону годов рождения. Написать функцию удаления сведений о дате рождения, если год рождения менее заданного числа. Написать функцию добавления информации о среднем результате спортсмена по всем соревнованиям.

6. Описать массив классов и поместить в него сгенерированные сведения о N деталях. Предусмотреть такие сведения как наименование детали, габаритные размеры (отдельный объект), материал, масса детали, список поставщиков деталей (отдельный массив из названий организации и контактного телефона). Написать функцию удаления сведений о материале, если масса детали менее указанной величины. Написать функцию добавления информации о габаритном объёме детали, найденного по габаритным размерам. Написать функцию выдачи списка деталей отсортированному по убыванию массы.

7. Описать массив классов и поместить в него сгенерированные сведения о N книгах. Предусмотреть такие сведения как название книги, жанр, дата издания (отдельный объект), количество экземпляров, ФИО автора (отдельным объектом), количество страниц. Написать функцию выдачи списка книг по фамилии автора, жанру или диапазону годов издания. Написать функцию удаления сведений о количестве страниц, если количество страниц менее заданного числа. Написать функцию добавления информации о возрасте книги, найденую по дате её издания.

8. Описать массив классов и поместить в него сгенерированные сведения о N сданных экзаменационных сессий. Предусмотреть такие сведения сессии как номер курса, дата начала сессии (отдельный объект), дата конца сессии (отдельный объект), список предметов (массив со сведениями о названии предмета и полученной оценки). Написать функцию удаления сведений о номере курса, если номер сессии нечётный. Написать функцию добавления информации о средней оценки сессии, найденного по списку оценок предметов. Написать функцию выдачи списка предметов и оценок лучшей сессии и худшей сессии.

9. Описать массив классов и поместить в него сгенерированные сведения о N работниках. Предусмотреть такие сведения как ФИО работника (отдельным объектом), дата рождения (отдельный объект), номер телефона, место работы (отдельный объект со сведениями о названии организации, должности и стаже). Написать функцию выдачи списка работников по названию организации, должности или диапазону стажа. Написать функцию удаления сведений о дате рождения, если стаж менее заданного числа. Написать функцию добавления информации о районе проживания работника, найденного по первым двум цифрам телефона.

10. Описать массив классов и поместить в него сгенерированные сведения о N сотрудниках. Предусмотреть такие сведения как ФИО сотрудника (отдельным объектом), дата рождения (отдельный объект), должность, стаж, зарплата (отдельный объект со сведениями о окладе, премии, оплате интенсивности, оплате переработки). Написать функцию удаления сведений о дате рождения, если стаж менее заданного числа. Написать функцию добавления информации о суммарном доходе работника, найденного как сумма всей составляющей зарплаты минус 13%. Написать функцию выдачи списка работников, отсортированных по убыванию дохода.

11. Описать массив классов и поместить в него сгенерированные сведения о N людях. Предусмотреть такие сведения как ФИО человека (отдельным объектом), пол, дата рождения (отдельный объект), номер телефона, адрес проживания (отдельный объект содержащий сведения о городе, улице, номере дома и номере квартиры). Написать функцию выдачи списка людей по городу, полу или диапазону годов рождения. Написать функцию удаления сведений о дате рождения, если год рождения более указанного. Написать функцию добавления информации о районе проживания работника, найденного по первым двум цифрам телефона.

12. «Массив бит» – BitArray. Разработать класс представляющий собой массив битов длины n. Реализовать функции для установки и получения значения бита на заданной позиции, изменения размера массива (справа и слева), сдвиг битов вправолево на заданное число позиций, битовые операции and и or для двух массивов. Предусмотреть функцию toString. Создать 2 массива классов и с помощью них поэлементно показать работу всех операций.

13. «Комплексное число» – Complex. Разработать класс комплексных чисел. Класс должен работать с функциями для изменения и получения значения действительной и мнимой части, для реализации операций сложения, вычитания, умножения, деления, присваивания комплексных чисел. Предусмотреть функцию toString. Создать 2 массива классов и с помощью них поэлементно показать работу всех операций.

14. «Односвязный список» – LinkedList. Разработать класс для работы с односвязным списком с целыми числами. Реализовать функции добавления элемента на заданную позицию, удаление всех элементов с заданным значением, получение значения по заданному индексу, объединение двух списков, разделение списка на два с указанной позиции, реверс списка. Предусмотреть функцию toString. Создать 2 массива классов и с помощью них поэлементно показать работу всех операций.

15. Описать массив классов и поместить в него сгенерированные сведения о плане выпуска N наименований. Предусмотреть такие сведения как название изделия, шифр, единица измерения, план выпуска (отдельный объект из плана выпуска и сколько фактически выпущено), список заказчиков (отдельный массив из названий организации и количества

закупаемого наименования). Написать функцию удаления сведений о единице измерения, если план выпуска менее заданного числа. Написать функцию добавления информации о проценте выполнения плана, найденного как соотношение фактического выпуска от плана выпуска. Написать функцию выдачи списка изделий, с перевыполнением плана, списка изделий с недовыполнением плана.

16. Описать массив классов и поместить в него сгенерированные сведения о N футболистах. Предусмотреть такие сведения как ФИО футболиста (отдельным объектом), дата рождения (отдельный объект), количество голов, команда (отдельный объект со сведениями о названии команды, стране, дате вступления (отдельный объект), зарплате футболиста). Написать функцию выдачи списка футболистов по названию команды, стране или диапазону забитых голов. Написать функцию удаления сведений о дате рождения, если год рождения менее заданного числа. Написать функцию добавления информации о количестве лет нахождения в команде, рассчитанной по году вступления в команду.

17. «Многочлен» – Polynom. Разработать класс полинома степени n. Реализовать функции для изменения и получения значения указанного коэффициента, вычисления значения полинома; сложения, вычитания, умножения полиномов. Предусмотреть функцию toString. Создать 2 массива классов и с помощью них поэлементно показать работу всех операций.

18. «Фигуры» – Shapes. Разработать класс для описания плоских фигур: круг, прямоугольник, треугольник. Включить функции для получения и изменения параметров фигур, перемещения на плоскости, вращения, нахождения площади и периметра фигуры. Предусмотреть функцию toString. Выполнить тестирование модуля, создав массив и показав на его примере работу всех функций.

19. «Дробь» – Fraction. Разработать класс в виде пары целых положительных чисел (m,n) а также отдельно знак дроби. Класс должен работать с функциями для изменения и получения значения числителя и знаменателя, сложения, вычитания, умножения, деления и присваивания дробей. Предусмотреть функцию toString. Создать 2 массива классов и с помощью них поэлементно показать работу всех операций.

20. Описать массив классов и поместить в него сгенерированные сведения о инвентаризационной ведомости из N наименований. Предусмотреть такие сведения как название наименования, инвентарный номер, дата принятия на учёт (отдельный объект), количество, место хранения (отдельный объект из полей номер корпуса, номер этажа, номер помещения). Написать функцию удаления сведений о дате принятия на учёт, если год принятия является текущим. Написать функцию добавления информации о сроке службы наименования по текущей дате и дате принятия его на учёт. Написать функцию выдачи списка наименований по номеру корпуса, номеру этажа или с указанным диапазоном сроков службы.

21. Описать массив классов и поместить в него сгенерированные сведения о N работниках. Предусмотреть такие сведения как ФИО работника (отдельным объектом), дата рождения (отдельный объект), номер цеха, трудовая информация (отдельный объект со сведениями о должности, разряде, стаже). Написать функцию выдачи списка работников по должности, разряду или диапазону стажа. Написать функцию удаления сведений о дате рождения, если стаж менее заданного числа. Написать функцию добавления информации о возрасте работника, найденного по году рождения.

22. Описать массив классов и поместить в него сгенерированные сведения о N студентах. Предусмотреть такие сведения как ФИО студента (отдельным объектом), дата поступления (отдельный объект), номер телефона, результаты сессии (отдельный массив с информацией о названии предметов и полученных оценках). Написать функцию удаления сведений о дате поступления, если год поступления старше заданного. Написать функцию добавления информации о среднем балле студента, найденного по оценкам сессии. Написать функцию выдачи списка студентов отсортированному по убыванию среднего балла.

23. «Квадратная матрица» – Matrix. Разработать класс квадратной матрицы n x n. Реализовать функции для изменения и получения значения элемента матрицы, сложения, вычитания, умножения матриц; вычисления индексов максимального и минимального элемента матрицы. Предусмотреть функцию toString. Создать 2 массива классов и с помощью них поэлементно показать работу всех операций.

24. «Бинарное дерево» – BinaryTree. Разработать класс для работы с бинарным деревом, узлы которого содержат натуральные числа. Реализовать функции добавления и удаления узлов, получения массива узлов с заданным значением, определения высоты и количества листьев у дерева. Предусмотреть функцию toString. Создать массив и с помощью них поэлементно показать работу всех операций.

4.4 Типовые темы курсовых работ

Тема 1. Проектирование обучающе-контролирующих программ.
Тема 2. Информационные системы
Тема 3. Игровые программы
Тема 4. Графические программы
Тема 5. Работа с матрицами. Рекомендуется для студентов заочной формы обучения.