Федеральное государственное автономное образовательное учреждение высшего образования «Московский физико-технический институт (национальный исследовательский университет)»

УТВЕРЖДЕНО
Проректор по учебной работе

А.А. Воронов

Рабочая программа дисциплины (модуля)

по дисциплине: Программирование на C++
по направлению: Информатика и вычислительная техника
профиль подготовки: Программная инженерия
передовая инженерная школа радиолокации, радионавигации и программной инженерии
кафедра информатики и вычислительной математики
курс: 1
квалификация: бакалавр

Семестр, формы промежуточной аттестации: 1 (осенний) - Дифференцированный зачет

Аудиторных часов: 120 всего, в том числе:
лекции: 30 час.
семинары: 60 час.
лабораторные занятия: 30 час.

Самостоятельная работа: 60 час.

Всего часов: 180, всего зач. ед.: 4

Количество контрольных работ, заданий: 4

Программу составил: В.Ю. Подаруев, канд. техн. наук, доцент

Программа обсуждена на заседании кафедры информатики и вычислительной математики 07.02.2022

Аннотация

В ходе данного курса обучающийся освоит классы, перегрузку операторов, полиморфизм, наследование, виртуальные функции, стандартную библиотеку STL, библиотеку Boost.

1. Цели и задачи

Цель дисциплины

Формирование базовых знаний по информатике для дальнейшего использования в других областях математического знания и дисциплинах естественнонаучного содержания; формирование информационной культуры, исследовательских навыков и способности применять знания на практике.

Задачи дисциплины

– Формирование у обучающихся базовых знаний по информатике;

– формирование информационной культуры: умение логически мыслить, проводить доказательства основных утверждений, устанавливать логические связи между понятиями;

– формирование умений и навыков применять полученные знания для решения информационных задач, самостоятельного анализа полученных результатов.

2. Перечень формируемых компетенций

Освоение дисциплины направлено на формирование следующих компетенций:

Код и наименование компетенции	Индикаторы достижения компетенции
ОПК-2 Способен использовать современные информационные технологии и программные средства при решении задач профессиональной деятельности, соблюдая требования информационной безопасности	ОПК-2.1 Способен применять современные вычислительную технику и сервисы сети Интернет в области (сфере) профессиональной деятельности
ПК-1 Способен ставить, формализовывать и решать задачи, в том числе разрабатывать и исследовать математические модели изучаемых явлений и процессов, системно анализировать научные проблемы, получать новые научные результаты	ПК-1.1 Способен находить, анализировать и обобщать информацию об актуальных результатах исследований в рамках тематической области своей профессиональной деятельности

3. Перечень планируемых результатов обучения по дисциплине (модулю)

В результате освоения дисциплины обучающиеся должны

знать:

- Основы дискретной математики;
- основы теории алгоритмов;
- свойства алгоритмов, проблемы алгоритмической сложности и алгоритмической неразрешимости;
- основы одного или нескольких алгоритмических языков программирования, общие характеристики языков программирования, идеологию объектно-ориентированного подхода;
- приемы разработки программ;
- общие понятия о структурах данных: стеки, очереди, списки, деревья, таблицы;
- основы архитектуры электронно-вычислительной машины (ЭВМ), представления информации в ЭВМ и архитектурные принципы повышения их производительности.

уметь:

- Выбирать оптимальные алгоритмы для современных программ;
- разрабатывать полные законченные программы на одном из языков программирования высокого уровня;
- разрабатывать программы на одном или нескольких языках программирования как индивидуально, так и в команде, с использованием современных средств написания и отладки программ;
- применять объектно-ориентированный подход для написания программ;
- использовать знания по информатике для приложения в инновационной, конструкторско-технологической и производственно-технологической сферах деятельности.

владеть:

- Одним или несколькими современными языками программирования и методами создания программ с использованием библиотек и современных средств их написания и отладки;
- навыками освоения современных архитектур ЭВМ.

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

4.1. Разделы дисциплины (модуля) и трудоемкости по видам учебных занятий

№	Тема (раздел) дисциплины	Трудоемкость по видам учебных занятий, включая самостоятельную работу, час.			
		Лекции	Семинары	Лаборат. работы	Самост. работа
1	Принципы объектно-ориентированного программирования	2	8	2	8
2	Адресное пространство приложения: динамические и статические переменные – члены класса	2	6	2	8
3	Перегрузка унарных и бинарных арифметических операторов. Инкапсуляция массивов объектов и перегрузка оператора индекса	6	8	2	8
4	Повторное использование классов. Наследование и перегрузка оператора присваивания объектов	6	6	4	8
5	Классы с виртуальными функциями	3	8	5	6
6	Шаблоны классов и шаблоны классов-контейнеров	4	8	5	8
7	Библиотеки STL и Boost	4	8	5	8
8	Динамическое идентификация и приведение типов	3	8	5	6
Итого часов		30	60	30	60
Подготовка к экзамену		0 час.			
Общая трудоёмкость		180 час., 4 зач.ед.			

4.2. Содержание дисциплины (модуля), структурированное по темам (разделам)

Семестр: 1 (Осенний)

1. Принципы объектно-ориентированного программирования

Понятие объекта, методы и члены класса, принципы ООП.

2. Адресное пространство приложения: динамические и статические переменные – члены класса

Адресное пространство приложения. Динамические члены класса. Динамическая память (куча) и статическая память (стек). Операции с динамической памятью (кучей). Конструкторы и деструкторы объектов в динамической памяти. Проблема утечки памяти в C++. Статические переменные – члены класса. Инициализация статических членов класса.

3. Перегрузка унарных и бинарных арифметических операторов. Инкапсуляция массивов объектов и перегрузка оператора индекса

Трансформирование базовых типов языка C++ в классы – основной источник формирования классов в языке C++. Классы-оболочки (Wrappers) с минимальным интерфейсом для базовых типов данных (примитивов). Полиморфизм конструкторов. Методы доступа. Ключевое слово this. Постоянные и модифицируемые члены класса. Массивы объектов. Классы-оболочки с полным интерфейсом инкапсулированного базового типа. Перегрузка операторов в C++. Инкапсуляция массивов базовых типов: абстрактные типы данных. Методы доступа. Перегрузка оператора индекса.

4. Повторное использование классов. Наследование и перегрузка оператора присваивания объектов

Повторное использование классов. Открытое наследование классов. Распространение и перегрузка наследуемых методов. Вызов конструкторов базового класса (суперкласса). Наследование функций и операций (Inheriting operations and functions). Перегрузка оператора присвоения.

5. Классы с виртуальными функциями

Множественное и виртуальное наследование. Закрытое наследование классов. Классы-адаптеры. Классы – композиты, в объекты которых вложены объекты других классов. Делегирование функций.

6. Шаблоны классов и шаблоны классов-контейнеров

Шаблоны. Полиморфизм функций и параметризованные функции (шаблоны). Параметризованные классы (шаблоны). Статический полиморфизм. Шаблоны интеллектуальных указателей.
Шаблоны классов-контейнеров. Вектор (массив). Строка. Связанный список. Обработка исключительных ситуаций. Внутренние и дружественные классы. Реализация двумерного вектора. Перегрузка оператора двойного индекса. Шаблоны итераторов для классов-контейнеров.

7. Библиотеки STL и Boost

Стандартная библиотека шаблонов STL (Standard Template Library) и библиотека шаблонов Boost. Классы-контейнеры и итераторы. Обобщенные алгоритмы. Функторы.

8. Динамическое идентификация и приведение типов

Чистые виртуальные функции и абстрактные классы. Полиморфное наследование абстрактных классов в C++. Производящие функции и фабрики объектов. Динамическое приведение типов и идентификация (RTTI) в Visual C++. Наследование классов с расширением интерфейса. Шаблоны оболочек-адаптеров и внешний полиморфизм.

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

Компьютерный класс с доской, проектором или телевизором, подключенный к сети.

6.Перечень рекомендуемой литературы

Основная литература

1. Язык программирования C++ [Текст] = The C++ Programming Language, [учеб. пособие для вузов] /Бьерн Страуструп ; пер. с англ. под ред. Н. Н. Мартынова. -М., БИНОМ, 2017

Дополнительная литература

1. Параллельное программировние многопоточных систем с разделяемой памятью [Текст] : учеб. пособие для вузов / А. Г. Тормасов .— М : Физматкнига, 2014 .— 208 с.

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

1. http://cs.mipt.ru
2. http://acm.mipt.ru

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

На лекциях используются мультимедийные технологии, включая демонстрацию презентаций.

Для контроля и коррекции знаний, обучающиеся могут использовать компьютерное тестирование, в том числе на сайте www.judge.mipt.ru.

В процессе самостоятельной работы обучающихся возможно использование любые среды программирования.

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

Студент, изучающий курс информатики, должен с одной стороны, овладеть общим понятийным аппаратом, а с другой стороны, должен научиться применять теоретические знания на практике.

Успешное освоение курса требует напряжённой самостоятельной работы студента. В программе курса приведено минимально необходимое время для работы студента над темой. Самостоятельная работа включает в себя:

– чтение и конспектирование рекомендованной литературы;

– проработку учебного материала (по конспектам лекций, учебной и научной литературе), подготовку ответов на вопросы, предназначенных для самостоятельного изучения, доказательство отдельных утверждений, свойств;

– решение задач, предлагаемых студентам на лекциях и лабораторных занятиях:

– подготовку к лабораторным занятиям, дифференцированному зачёту.

Руководство и контроль за самостоятельной работой студента осуществляется в форме индивидуальных консультаций.

Показателем владения материалом служит умение решать задачи. Для формирования умения применять теоретические знания на практике студенту необходимо решать как можно больше задач. При решении задач каждое действие необходимо аргументировать, ссылаясь на известные теоретические сведения.

При подготовке к лабораторным занятиям необходимо повторять ранее изученные основные понятия. В начале занятия, как правило, проводится короткий (10-15 минут) опрос по материалу прошедших занятий в устной или письменной форме. Обычно придерживаются следующей схемы: изучение материала лекции по конспекту в тот же день, когда была прослушана лекция (10-15 минут); повторение материала накануне следующей лекции (10-15 минут), проработка учебного материала по конспектам лекций, учебной и научной литературе, подготовка ответов на вопросы, предназначенных для самостоятельного изучения (1 час неделю), подготовка к практическому занятию, решение задач (1 час). Важно добиться понимания изучаемого материала, а не механического его запоминания. При затруднении изучения отдельных тем, вопросов, следует обращаться за консультациями к лектору или преподавателю, ведущему лабораторные занятия.

Обязательным требованием является выполнение домашних работ, которые оформляются в специально отведённой для этого тетради и систематически сдаются на проверку.

Промежуточный контроль знаний проводится в виде контрольных работ, на которых студенту предлагается решить несколько задач, а также студенту в ходе освоения курса необходимо выполнить две домашние индивидуальные работы с их последующей защитой.

ПРИЛОЖЕНИЕ

ОЦЕНОЧНЫЕ МАТЕРИАЛЫ ПО ДИСЦИПЛИНЕ (МОДУЛЮ)

по направлению: Информатика и вычислительная техника
профиль подготовки: Программная инженерия
передовая инженерная школа радиолокации, радионавигации и программной инженерии
кафедра информатики и вычислительной математики
курс: 1
квалификация: бакалавр

Семестр, формы промежуточной аттестации: 1 (осенний) - Дифференцированный зачет

Разработчик: В.Ю. Подаруев, канд. техн. наук, доцент

1. Компетенции, формируемые в процессе изучения дисциплины

Код и наименование компетенции	Индикаторы достижения компетенции
ОПК-2 Способен использовать современные информационные технологии и программные средства при решении задач профессиональной деятельности, соблюдая требования информационной безопасности	ОПК-2.1 Способен применять современные вычислительную технику и сервисы сети Интернет в области (сфере) профессиональной деятельности
ПК-1 Способен ставить, формализовывать и решать задачи, в том числе разрабатывать и исследовать математические модели изучаемых явлений и процессов, системно анализировать научные проблемы, получать новые научные результаты	ПК-1.1 Способен находить, анализировать и обобщать информацию об актуальных результатах исследований в рамках тематической области своей профессиональной деятельности

2. Показатели оценивания компетенций

В результате изучения дисциплины «Программирование на С++» обучающийся должен:

знать:

- Основы дискретной математики;
- основы теории алгоритмов;
- свойства алгоритмов, проблемы алгоритмической сложности и алгоритмической неразрешимости;
- основы одного или нескольких алгоритмических языков программирования, общие характеристики языков программирования, идеологию объектно-ориентированного подхода;
- приемы разработки программ;
- общие понятия о структурах данных: стеки, очереди, списки, деревья, таблицы;
- основы архитектуры электронно-вычислительной машины (ЭВМ), представления информации в ЭВМ и архитектурные принципы повышения их производительности.

уметь:

- Выбирать оптимальные алгоритмы для современных программ;
- разрабатывать полные законченные программы на одном из языков программирования высокого уровня;
- разрабатывать программы на одном или нескольких языках программирования как индивидуально, так и в команде, с использованием современных средств написания и отладки программ;
- применять объектно-ориентированный подход для написания программ;
- использовать знания по информатике для приложения в инновационной, конструкторско-технологической и производственно-технологической сферах деятельности.

владеть:

- Одним или несколькими современными языками программирования и методами создания программ с использованием библиотек и современных средств их написания и отладки;
- навыками освоения современных архитектур ЭВМ.

3. Перечень типовых (примерных) вопросов, заданий, тем для подготовки к текущему контролю

С целью контроля освоения обучающимися учебного материала проводится устный опрос в начале занятия по теме прошлого занятия.

3. Перечень типовых заданий, используемых для оценки знаний, умений, навыков

Промежуточная аттестация по дисциплине «Программирование на С++» осуществляется в форме дифференцированного зачета. Дифференцированный зачет выставляется по результатам работы студента в течение семестра на основе оценок за лабораторные работы, домашние задания, контрольные работы.

Вопросы к зачету.

1 (осенний) семестр

Сравнение синтаксиса языка С и С++. Перегрузка имен (полиморфизм) функций. Константы времени компиляции и статическая типизация в объектно-ориентированных языках программирования (по Барбаре Лисков). Структуры в С++. Синтаксис классов изолированных (невзаимодействующих) объектов в языке С++. Классы, представляющие нормативное знание. Конструкторы и деструкторы. Интерфейс и его реализация.

Трансформирование базовых типов языка С++ в классы – основной источник формирования классов в языке С++. Классы-оболочки (Wrappers) с минимальным интерфейсом для базовых типов данных (примитивов). Полиморфизм конструкторов. Методы доступа. Ключевое слово this. Постоянные и модифицируемые члены класса. Массивы объектов. Классы-оболочки с полным интерфейсом инкапсулированного базового типа. Перегрузка операторов в С++. Инкапсуляция массивов базовых типов: абстрактные типы данных. Методы доступа. Перегрузка оператора индекса.

Дружественные функции: функции, дружественные классам – функции за пределами классов (нарушение концепции объектно-ориентированного программирования). Инкапсуляция дружественных функций в класс Visitor. Объектно-ориентированное программирование без использования дружественных функций: классы данных и классы функций. Потоки ввода-вывода данных в С++. Форматируемый ввод/вывод и манипуляторы ввода/вывода. Перегрузка пользовательских операторов ввода/вывода как дружественных функций. Файловый ввод/вывод. Неформатируемый двоичный ввод/вывод (байтовые потоки). Инкапсуляция объектов и вертикальное делегирование функций.

Адресное пространство приложения. Динамические члены класса. Динамическая память (куча) и статическая память (стек). Операции с динамической памятью (кучей). Конструкторы и деструкторы объектов в динамической памяти. Проблема утечки памяти в С++. Статические переменные – члены класса. Инициализация статических членов класса.

Копирование объектов. Присваивание объектов. Передача и возвращение функциями объектов по значению. Конструкторы копий. Клонирование объектов в С++. Передача и возвращение ссылок на объекты.

Инкапсуляция массивов объектов. Классы-оболочки контейнеров с расширенным интерфейсом. Специальные методы создания объектов. Создание объектов посредством статических методов. Класс Singleton. Косвенное создание объектов и горизонтальное делегирование функций. Класс-заместитель Proxy. Классы-оболочки для указателей базовых типов. Перегрузка операций указывания и разыменования. Интеллектуальные указатели.

Повторное использование классов. Открытое наследование классов. Распространение и перегрузка наследуемых методов. Вызов конструкторов базового класса (суперкласса). Наследование функций и операций (Inheriting operations and functions). Перегрузка оператора присвоения. Множественное и виртуальное наследование. Закрытое наследование классов. Классы-адаптеры. Классы – композиты, в объекты которых вложены объекты других классов. Делегирование функций.

Шаблоны. Полиморфизм функций и параметризованные функции (шаблоны). Параметризованные классы (шаблоны). Статический полиморфизм. Шаблоны интеллектуальных указателей.

Шаблоны классов-контейнеров. Вектор (массив). Строка. Связанный список. Обработка исключительных ситуаций.

Внутренние и дружественные классы. Реализация двумерного вектора. Перегрузка оператора двойного индекса. Шаблоны итераторов для классов-контейнеров.

Стандартная библиотека шаблонов STL (Standard Template Library). Классы-контейнеры и итераторы. Обобщенные алгоритмы. Функторы.
Классы с виртуальными функциями. Подстановочный критерий Барбары Лисков. Структура объектов и таблицы виртуальных функций. Динамический полиморфизм.
Чистые виртуальные функции и абстрактные классы. Полиморфное наследование абстрактных классов в C++. Производящие функции и фабрики объектов.
Динамическое приведение типов и идентификация (RTTI) в Visual C++.
Наследование классов с расширением интерфейса. Шаблоны оболочек-адаптеров и внешний полиморфизм.
Одновременное (конкурентное) выполнение потоков вычислений.

1 (осенний) семестр

Задание 1

Списки

Реализовать класс List для создания структуры данных двухсвязного списка, каждый элемент которого Node хранит уникальный, случайно сгенерированный ключ.
Класс должен содержать методы:

- добавление элемента в начало и конец списка (*аргументы*: значение ключа для нового узла);
- вывод списка на экран;
- поиск элемента по значению ключа (*аргументы*: значение ключа);
- удаление элемента из начала списка, из конца списка, а также по значению ключа (*аргументы*: значение ключа);
- поиск количества элементов.

Программа должна создать заданный пользователем список и продемонстрировать работоспособность всех методов класса.

Бинарные деревья

Реализовать класс Tree для создания структуры данных бинарного дерева, каждый элемент которого Node хранит уникальный, случайно сгенерированный ключ.
Класс должен содержать методы:

- добавление узла (*аргументы*: значение ключа для нового узла);
- вывод дерева на экран;
- поиск узла дерева по значению ключа (*аргументы*: значение ключа);
- удаление узла по значению ключа (*аргументы*: значение ключа);
- поиск максимального элемента дерева;
- поиск суммы всех элементов дерева;
- поиск количества узлов дерева;
- поиск максимальной глубины дерева.

Программа должна создать дерево заданного размера и продемонстрировать работоспособность всех методов класса.

Векторы и матрицы

Реализовать класс Vector3 для трехмерных векторов. Перегрузить для него основные арифметические операции, в том числе:

- умножение на константу;
- сложение;
- вычитание;
- скалярное и векторное умножение.

Реализовать класс Matrix3 для матриц 3x3. Перегрузить для него основные арифметические операции:

- сложение;
- вычитание;
- определитель;

- умножение на вектор;
- умножение двух матриц.

Аналитическая геометрия

С использованием класса Vector реализовать класс Line, Plane, Segment и Triangle для прямой, плоскости, отрезка и треугольника соответственно. Реализовать класс CollisionDetector, который умеет находить пересечение любых двух объектов указанных типов (отрезок с отрезком, отрезок с плоскостью, треугольник с плоскостью и т.д.).

Задание 2

Геометрические объекты

Реализовать класс Figure, содержащую общую информацию о геометрическом объекте и основные методы работы с ним:

- положение в пространстве;
- поворот;
- перемещение;
- растяжение вдоль осей.

Реализовать набор классов, наследованных от Figure (и, возможно, друг от друга): Triangle (треугольник), Sphere (сфера), Ellipsoid (эллипсоид), Cube (куб), Parallelepiped (параллелепипед), …

Реализовать в классе Figure статический метод CollisionDetector, который умеет определять, пересекаются ли два геометрических объекта.

Реализовать в классе Figure счетчик созданных объектов.

«Дождь»

В замкнутом кубическом объеме на верхней границе случайно генерируются различные объекты типа Figure с произвольной начальной скоростью, направленной вниз. После пересечения нижней границы объекты исчезают. Одновременно в объеме должно быть порядка 1000 объектов.

Программа должна работать без утечек памяти, проверить это можно при помощи valgrind.

Дополнительное задание. Визуализация процесса. Максимально увеличить количество объектов, которые могут одновременно находиться в объеме.

Бинарные деревья поиска

На основе класса Tree из первого задания реализовать класс SearchTree для создания структуры данных бинарного дерева поиска.

Универсальный список

На основе класса List из первого задания реализовать аналог list из STL.

Моделирование

Дополнительное задание ко всем задачам данного раздела – визуализация процесса, описанного в задаче.

«Задача трех тел»

В неограниченном трехмерном пространстве находится N шарообразных объектов различной массы и различного радиуса. В начальный момент времени они имеют случайные (либо заданные пользователем) векторы скоростей. Объекты взаимодействуют друг с другом согласно закону всемирного тяготения. Определить, какие из объектов столкнутся, и время столкновения.

Дополнительное задание. Вводится поправка к закону всемирного тяготения, которая обеспечивает диссипацию энергии – например, если спутник находился на стабильной орбите вокруг планеты, после введения поправки он будет на нее постепенно падать. Необходимо разработать модель этой поправки (например, приливные силы или гравитационные волны) и ввести ее в программу.

«Тренировка футболистов»

Команда футболистов отрабатывает индивидуальные действия. Каждый играет за себя. Цель футболиста – завладеть мячом и забить гол. Таким образом, он может выполнять два действия:

1. бороться за мяч.
2. бить по воротам, как только мяч оказался у него.

После удара по воротам вратарь выбивает мяч в произвольное место поля. Футболист забивает гол с определенной вероятностью, зависящей от расстояния до ворот. Команда играет определенное время, потом определяется победитель – по забитым мячам. Вывести на экран счет и последовательность действий каждого футболиста.

«Ветеран Броуновского движения»

В кубическом замкнутом объеме находится N «маленьких» шарообразных частиц массы m и одна «большая» массы M. В начальный момент времени они имеют случайные векторы скоростей. Частицы упруго отталкиваются друг от друга и от стенок. Построить график зависимости скорости «большой» частицы от времени.

«Воробьи»

Бабушка на скамейке периодически (пусть раз в Q минут) бросает воробьям по одной ровно N хлебных крошек. Когда бросают крошку хлеба, первый подлетевший к ней воробей хватает крошку, относит ее в сторону и съедает. За счет этого он пропускает «розыгрыш» следующей крошки. Первоначально у скамейки находится M воробьев. Каждую минуту к стае прилетает еще один воробей. Съев P крошек, воробей наедается и улетает. Чтобы количество воробьев не увеличивалось, должно выполняться соотношение $Q = N/P$.

Итак, воробей может выполнять действия:

2. Прилететь.
3. Ждать раздачи крошек.
4. Драться за крошку.
5. Отлететь с крошкой в сторону и съесть ее.
6. Улететь.

Вывести на экран последовательность действий каждого из воробьев.

4. Критерии оценивания

За каждый вопрос из контрольного задания студент получает от 0 до максимального балла в зависимости от полноты представленного ответа (решения). Критерии проставления баллов утверждаются на заседании учебно-методической комиссии кафедры. Процентсуммарно набранных баллов от максимально возможного количества определяет оценку за теоретические знания по каждому контрольному заданию:

Оценка	Набранные баллы
отлично (10)	более 88%
отлично (9)	от 78% до 88% включительно
отлично (8)	от 68% до 78% включительно
хорошо (7)	от 58% до 68% включительно
хорошо (6)	от 48% до 58% включительно
хорошо (5)	от 38% до 48% включительно
удовлетворительно (4)	от 28% до 38% включительно
удовлетворительно (3)	от 18% до 28% включительно
неудовлетворительно (2)	от 08% до 18% включительно
неудовлетворительно (1)	не более 08%

Каждая лабораторная работа и задача из задания при полном правильном решении оценивается в определенное количество баллов от 5 до 25. Баллы за полностью правильное решение определяются преподавателями и утверждаются на заседании учебно-методической комиссии кафедры. Процентсуммарно набранных баллов от

максимально возможного количества определяет оценку за практические знания студента по вышеприведенной таблице.

5. Методические материалы, определяющие процедуры оценивания знаний, умений, навыков и (или) опыта деятельности

Время проведения каждой контрольной работы составляет 2 академических часа.

Во время проведения контрольных работ обучающиеся могут пользоваться программой дисциплины и любыми рукописными материалами.