Образовательная автономная некоммерческая организация

высшего образования

«МОСКОВСКИЙ ТЕХНОЛОГИЧЕСКИЙ ИНСТИТУТ»

Актуализированная версия
утверждена на заседании
Ученого совета
ОАНО ВО «МосТех»
протокол № 07 от 12 февраля 2026 г.

УТВЕРЖДАЮ
Ректор
_____Ю.В. Вепринцева

«12» февраля 2026 г.

РАБОЧАЯ ПРОГРАММА УЧЕБНОГО ПРЕДМЕТА

«СИНТАКСИС С++»

(наименование программы)

ДОКУМЕНТ ПОДПИСАН
ЭЛЕКТРОННОЙ ПОДПИСЬЮ

Сертификат: 02C18481005DB3E89B4ECFE9757A453BC1
Владелец: УСАЧЁВ ПАВЕЛ АЛЕКСАНДРОВИЧ
Действителен: с 19.09.2025 до 19.12.2026

Москва 2026

ДОКУМЕНТ ПОДПИСАН
ЭЛЕКТРОННОЙ ПОДПИСЬЮ

Сертификат: 501FF1003FB3EB8C4EDC9ADF471C38C4
Владелец: Вепринцева Юлия Владимировна
Действителен: с 20.08.2025 до 20.08.2026

1. Цель изучения учебного предмета

Обучение синтаксису языка программирования C++.

2. Планируемые результаты обучения по учебному предмету

Знать	• Синтаксис языка C++, особенности программирования на этом языке, стандартные библиотеки языка программирования; • Методологии разработки компьютерного программного обеспечения.
Уметь	• Применять язык C++ для написания программного кода.
Владеть	• Навыком создания программного кода в соответствии с техническим заданием (готовыми спецификациями).

. Содержание учебного предмета

3.1. Распределение учебного времени, выделенного на контактную работу обучающихся с преподавателем, на самостоятельную работу обучающихся и учебные часы с использованием дистанционных образовательных технологий

Общая трудоемкость (объем) учебного предмета составляет 36 академических часов.

	Наименование тем учебного предмета	Общая трудоемкость, ч.	Всего, ч.	Контактная работа, ч			Учебные занятия с применением дистанционных технологий, ч	Самостоятельная работа, ч	Форма аттестации
				Лекции	Лабораторные работы	Практические, семинарские занятия и др. занятия			
1	2	3	4	5	6	7	8	9	10
1.	Тема 1. “Введение”	2	0	0	0	0	1	1	-
2.	Тема 2. “Переменные”	4	0	0	0	0	2	2	-
3.	Тема 3. “Условный оператор и циклы”	4	0	0	0	0	2	2	-
4.	Тема 4. “Массивы”	6	0	0	0	0	3	3	-
5.	Тема 5. “Функции”	6	0	0	0	0	3	3	-
6.	Тема 6. “Сапёр”	4	0	0	0	0	2	2	-
7.	Тема 7. “Файлы”	3	0	0	0	0	2	1	-
8.	Тема 8. “Исключения”	2	0	0	0	0	1	1	-
9.	Тема 9. “Указатели”	3	0	0	0	0	2	1	-
	Промежуточная аттестация	2	0	0	0	0	0	0	Зачёт
	Итого:	36	0	0	0	0	18	16	2

3.2. Содержание тем учебного предмета

Тема 1. “Введение”

- Введение

Тема 2. “Переменные”

- Переменные
- Операции над переменными

Тема 3. “Условный оператор и циклы”

- Условный оператор
- Циклы

Тема 4. “Массивы”

- Массивы
- Двумерные массивы

Тема 5. “Функции”

- Функции
- Рекурсивные функции

Тема 6. “Сапёр”

- Сапёр

Тема 7. “Файлы”

- Файлы

Тема 8. “Исключения”

- Исключения

Тема 9. “Указатели”

- Указатели
- Динамический массив, список

4. Учебно-методическое и информационное обеспечение учебного предмета

№ п/п	Вид и наименование литературы
	Основная
1.	Лебеденко, Л. Ф. Основы программирования на C++ : учебное пособие : [16+] / Л. Ф. Лебеденко, О. И. Моренкова. – 2-е изд., перераб. и доп. – Новосибирск : Сибирский государственный университет телекоммуникаций и информатики, 2021. – 200 с. : ил., табл., схем. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=694769
2.	Информационные технологии и программирование : учебное пособие : [16+] / Е. В. Булгакова, А. Н. Кубанков, М. Д. Хананашвили, Д. С. Дойников. – Москва ; Вологда : Инфра-Инженерия, 2025. – 120 с. : ил., табл. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=725648
3.	Исаева, Г. Н. Языки программирования : практикум по курсу «Языки программирования» : учебное пособие : [16+] / Г. Н. Исаева, Н. В. Логачёва, Ю. В. Стреналюк ; Технологический университет. – Москва : Директ-Медиа, 2024. – 109 с. : ил., табл. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=713440
	Дополнительная
4.	Карчевская, М. П. Основы алгоритмизации инженерных задач : учебное пособие : [16+] / М. П. Карчевская, О. Л. Рамбургер. – Москва ; Вологда : Инфра-Инженерия, 2025. – 244 с. : ил., табл. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=725669
5.	Шапкин, А. С. Задачи с решениями по высшей математике, теории вероятностей, математической статистике, математическому программированию : учебное пособие / А. С. Шапкин, В. А. Шапкин. – 11-е изд., перераб. – Москва : Дашков и К°, 2024. – 402 с. : ил., табл., схем. – (Учебные издания для бакалавров). – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=720215
	Ресурсы информационно-коммуникационной сети «Интернет»
1.	Электронная библиотека на платформе https://lms.mti.moscow/ - https://biblioclub.ru/

5. Учебно-материальная база, необходимая для осуществления образовательного процесса по учебному предмету

Материально-техническое обеспечение учебных предметов включает в себя:

- Персональный компьютер/мобильное устройство (обучающийся обеспечивает себе самостоятельно) с любой операционной системой, позволяющей использовать браузеры и подключаться к сети «Интернет»;
- Рекомендовано установить на персональный компьютер стандартный пакет офисных программ и программ необходимых для выполнения самостоятельной работы (полный список возможно уточнить у куратора программы);
- Обеспечение доступа электронную информационно-образовательную среду Института - lms.mti.moscow;
- Обеспечение доступа в электронную библиотеку biblioclub.ru.

6. Методические рекомендации (указания, материалы) для преподавателей и обучающихся

В процессе изучения данной учебного предмета используются следующие виды учебных занятий и работ: учебные занятия с применением дистанционных образовательных технологий, самостоятельная работа.

6.1 Методические указания для преподавателей при проведении занятий с применением ДОТ.

В процессе занятий с применением ДОТ рекомендуется вести конспект, что позволит впоследствии вспомнить изученный теоретический и практический учебный материал, выполнить самостоятельную работу и подготовиться к итоговой аттестации.

Желательно оставить в рабочих конспектах поля, на которых делать пометки из рекомендованной литературы, дополняющие материал пройденного занятия с применением ДОТ, а также подчеркивающие особую важность тех или иных положений.

На практических занятиях для приобретения умений и навыков целесообразно последовательно выполнять действия за преподавателем, который демонстрирует решение практических заданий, кейсов и т.п.

В завершении занятия с применением ДОТ преподаватель знакомит обучающихся с литературой (основной, дополнительной), с практическими заданиями для самостоятельной работы и даёт рекомендации по их выполнению. Полученную информацию целесообразно кратко и лаконично записывать.

6.2 Методические указания для обучающихся при обучении в виде занятий с применением ДОТ.

В процессе занятий с применением ДОТ рекомендуется вести конспект, что позволит впоследствии вспомнить изученный учебный материал, выполнить самостоятельную работу и подготовиться к промежуточной аттестации по учебному предмету.

Желательно оставить в рабочих конспектах поля, на которых делать пометки из рекомендованной литературы, дополняющие материал прослушанного занятия с применением ДОТ.

Занятия с применением ДОТ имеют логическое завершение, роль которого выполняет заключение. Также в завершении занятия с применением ДОТ преподаватель знакомит обучающихся с литературой (основной, дополнительной), с практическими заданиями для самостоятельной работы и даёт рекомендации по их выполнению. Полученную информацию целесообразно кратко и лаконично записывать.

6.3. Методические указания для обучающихся по выполнению самостоятельной работы

Самостоятельная работа является обязательной для каждого обучающегося, ее объем по учебному предмету определяется учебным планом. Самостоятельная работа предполагает изучение обязательной и дополнительной литературы и выполнение практических заданий, направленных на достижение планируемых результатов по учебному предмету. Практические задания выполняются последовательно с изучением тем учебного предмета и получения знаний в процессе занятий.

6.4. Методические указания для преподавателей по организации самостоятельной работы

Для организации самостоятельной работы обучающихся преподаватель готовит перечень обязательной и дополнительной литературы, а также практические задания, направленные на достижение планируемых результатов по учебному предмету.

Практические задания для самостоятельной работы:

1. **Задание**

Вы познакомились с переменными, в том числе узнали как объявить переменную, задать ей значение через консоль и вывести это значение обратно в консоль. Для отработки данного навыка создайте несколько переменных основных типов, введите их значения через консоль и после выведите все сохранённые значения обратно в консоль.

1. Объявите переменные различных типов: int, double, char, bool и string.
2. Запросите у пользователя ввод значения для каждой переменной и сохраните эти значения в соответствующие переменные с помощью std::cin
3. Выведите все значения переменных через пробел обратно в консоль.
4. В итоге должна получиться следующая программа:
5. Запустите её, нажав сочетание клавиш Ctrl + F5 или вот эту кнопку:

У вас откроется консоль, где нужно будет по очереди вводить подходящие значения переменных и нажимать клавишу Enter

Протестируйте программу.

2. **Задание**

Вы узнали как работает условный оператор. Для отработки данного навыка напишите программу, которая определяет чётность числа.

1. Объявите переменную типа int
2. Запросите у пользователя ввод целого числа и сохраните его в соответствующей переменной
3. Проверьте остаток от деления введённого числа на 2 с помощью условного оператора
4. Если остаток равен 0, то число чётное. Выведите соответствующее сообщение.
5. Если остаток не равен 0, то число нечётное. Выведите соответствующее сообщение.
6. В итоге должна получиться следующая программа:

Протестируйте программу несколько раз.

3. **Задание**

Написать программу, которая создаёт текстовый файл, записывает в него строку, введённую пользователем в консоль. Затем программа должна закрыть файл для записи, открыть его для чтения, считать записанную строку и вывести её в консоль.

4. **Задание**

Создать целочисленную переменную, а также указатель ну эту целочисленную переменную. Вывести в консоль адрес переменной двумя способами (используя саму переменную и указатель на переменную). Также вывести в консоль значение переменной, используя указатель на неё.

5. **Задание**

Пользователь с клавиатуры вводит два целых числа N и M — количество строк и столбцов в матрице (оба числа больше 0).

Программа должна:

- создать матрицу размером N x M;
- заполнить её случайными двузначными числами (от 10 до 99);
- аккуратно вывести матрицу в консоль в виде таблицы (каждая строка с новой строки, элементы разделены пробелами или табуляцией).

После этого:

- пользователь вводит, что он хочет посчитать: сумму элементов строки или столбца (можно, например, сначала спросить: 1 – сумма строки, 2 – сумма столбца);
- затем пользователь вводит номер выбранной строки или столбца (с учётом, что нумерация начинается с 1);
- программа вычисляет и выводит сумму элементов указанной строки или указанного столбца.

При неверном вводе (номер вне диапазона) программа должна сообщать об ошибке.

6. Задание

Напишите рекурсивную функцию, которая переводит число из десятичной системы в двоичную. Допускается использовать тип string для хранения двоичного числа.

7. Задание

Реализуйте программу «Заметки».

При запуске программы:

- определяется директория, в которой находится сама программа;
- выводится список доступных заметок — имена всех текстовых файлов (например, с расширением .txt) в этой директории.

После вывода списка пользователь должен выбрать действие:

1. Открыть существующую заметку:
   - пользователь вводит номер или имя файла из списка;
   - программа открывает выбранный текстовый файл;
   - считывает его содержимое и выводит текст заметки в консоль.
2. Создать новую заметку:
   - пользователь вводит имя нового файла (например, note1.txt);
   - затем вводит текст заметки (можно до пустой строки или специального символа окончания ввода);
   - программа записывает введённый текст в новый файл в директории программы;
   - при успешном создании можно вывести сообщение, что заметка сохранена.

При некорректном выборе (несуществующий файл, пустое имя и т.п.) программа должна вывести понятное сообщение об ошибке.

8. Задание

8. Реализовать упрощённый вариант стека — структуры данных, в которой доступен только последний добавленный элемент (принцип LIFO: Last In — First Out).

Требования:

- Элемент стека должен быть структурой, которая хранит:
  - целое число (данные);
  - указатель на следующий элемент стека (обычно на предыдущий добавленный элемент).

  Работа программы:

1. Пользователь вводит количество элементов стека.
2. Затем пользователь по одному вводит значения (целые числа) для каждого элемента.

   При каждом вводе:
   - создаётся новый элемент структуры;
   - он добавляется в стек как «верхний» элемент (push).
3. После ввода всех элементов программа должна вывести элементы стека в правильном порядке снятия:
   - сначала выводится последний добавленный элемент,
   - затем предпоследний и так далее до самого первого (реализация операции pop).
4. После завершения работы со стеком необходимо корректно удалить все элементы:
   - последовательно извлекать элементы из стека;
   - освобождать память, выделенную под каждый элемент (если используется язык с ручным управлением памятью, например C/C++).

7. Оценочные материалы для промежуточной аттестации обучающихся по учебному предмету

Форма проведения промежуточной аттестации – зачет.

Зачет проводится в виде электронного тестирования.

Общее время, требуемое на выполнение – 2 часа.

Порядок проведения электронного тестирования: проводится дистанционно, предполагает прохождение обучающимися теста, направленного на оценку сформированности планируемых результатов обучения, состоящего из 25 вопросов. Каждый правильный ответ оценивается в 4 балла, неправильный оценивается в 0 баллов.

Перечень вопросов, выносимых на электронное тестирование:

1. Что такое C++?
    a. машинный язык
    b. высокоуровневый язык программирования общего назначения
    c. только язык для веб-разработки
2. Какой вариант подключения стандартной библиотеки ввода-вывода корректен?
    a. #include <iostream>
    b. include <iostream>
    c. #include iostream
3. Как правильно объявить точку входа в консольное приложение на C++?
    a. void main()
    b. int main()
    c. main()
4. Как правильно объявить целочисленную переменную x со значением 10?
    a. int x = 10;
    b. x int = 10;
    c. int x == 10;
5. Какой тип данных используется для хранения числа с плавающей точкой двойной точности?
    a. float
    b. double
    c. char
6. Какой оператор используется для присваивания значения переменной?
    a. ==
    b. =
    c. : =
7. Чему равно выражение 7 % 3 в C++?
    a. 1
    b. 2
    c. 0
8. Что делает оператор ++x (префиксный инкремент)?
    a. увеличивает x на 1 и возвращает новое значение
    b. увеличивает x на 1 и возвращает старое значение
    c. ничего не делает
9. Какой синтаксис условного оператора if в C++ верный?

a. if (условие) оператор;
b. if условие: оператор
c. if условие then оператор;

10. Какой оператор используется для альтернативной ветки без условия в конструкции if?
    a. else if
    b. elseif
    c. else
    d. оператор увеличения счётчика цикла i++
11. Какой цикл в C++ выполняется хотя бы один раз независимо от условия?
    a. for
    b. while
    c. do ... while
12. Какой заголовок цикла верно задаёт вывод чисел от 0 до 4?
    a. for (int i = 0; i < 5; i++)
    b. for (int i = 0; i <= 5; i++)
    c. for (int i = 1; i < 5; i++)
13. Выберите истинное утверждение об инициализации массива
    a. массив при объявлении должен быть полностью инициализирован, иначе компилятор выдаст ошибку
    b. при объявлении обязательно указывается размер массива, а инициализировать элементы некоторыми значениями можно позже
    c. пользователь сам в процессе выполнения программы может задать размер массива и провести его инициализацию
14. Как объявить массив из 5 целых чисел в C++?
    a. int a(5);
    b. int a[5];
    c. int[5] a;
15. Как объявить двумерный массив 3×4 целых чисел?
    a. int a[3][4];
    b. int a[4][3];
    c. int a(3,4);
16. Как объявить функцию sum, которая принимает два int и возвращает int?
    a. int sum(int a, int b) { ... }
    b. void sum(int a, int b) { ... }
    c. sum int(int a, int b) { ... }
17. Что такое рекурсивная функция?
    a. функция, которая вызывает сама себя
    b. функция, которая вызывается только из main
    c. функция, которая не имеет параметров
18. Какой тип удобно использовать для хранения информации о наличии мины в ячейке (мина / нет мины)?
    a. bool
    b. double
    c. string
19. Какой заголовочный файл нужен для работы с файловыми потоками C++ (ifstream, ofstream)?

a. #include <fstream>
b. #include <file>
c. #include <stdio.h>

20. Как открыть файл data.txt для записи с помощью ofstream?
    a. ofstream file("data.txt");
    b. open ofstream("data.txt");
    c. file = ofstream;
21. Что позволяет сделать отладчик в Visual Studio?
    a. автоматически проверить программу на наличие ошибок
    b. остановить программу в любой момент выполнения
    c. пошагово выполнить программу, отслеживая значения переменных
22. Какой синтаксис блока обработки исключений в C++ корректен?
    a. try { /__код__/ } catch (int e) { /__обработка__/ }
    b. try: /__код__/ catch (int e): /__обработка__/
    c. exception (int e) { /__обработка__/ }
23. Как правильно объявить указатель на целое число в C++?
    a. int __p;
    b. int p__;
    c. *int p;
24. Как правильно выделить динамический массив из 10 целых чисел?
    a. int a[10] = new int;
    b. int* a = new int[10];
    c. int* a = new int(10);
25. Что такое файл?
    a. выделенная для записи или чтения область на жёстком диске
    b. иконка на рабочем столе, позволяющая запустить ту или иную программу
    c. именованный набор байтов, который может быть сохранён на жёстком диске или другому накопителе

Критерии оценивания промежуточной аттестации:

Перевод полученных баллов по результатам тестирования и отчеты о выполнении практических заданий в отметку производится следующим образом:

90 -100 баллов «отлично»/ «зачтено»;

70 - 89 баллов «хорошо»/ «зачтено»;

50 - 69 баллов «удовлетворительно»/ «зачтено»;

менее 50 баллов «неудовлетворительно»/ «не зачтено».

Образовательная автономная некоммерческая организация

высшего образования

«МОСКОВСКИЙ ТЕХНОЛОГИЧЕСКИЙ ИНСТИТУТ»

Актуализированная версия
утверждена на заседании
Ученого совета
ОАНО ВО «МосТех»
протокол № 07 от 12 февраля 2026 г.

УТВЕРЖДАЮ
Ректор
_____Ю.В. Вепринцева

«12» февраля 2026 г.

РАБОЧАЯ ПРОГРАММА УЧЕБНОГО ПРЕДМЕТА «ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ»

(наименование программы)

Москва 2026

Знать	• Методологии и технологии объектно-ориентированного программирования; • Технологии программирования.
Уметь	• Использовать выбранную среду для объектно-ориентированного программирования.

3. Содержание учебного предмета

3.1. Распределение учебного времени, выделенного на контактную работу обучающихся с преподавателем, на самостоятельную работу обучающихся и учебные часы с использованием дистанционных образовательных технологий

Общая трудоемкость (объем) учебного предмета составляет 30 академических часов.

Наименование тем учебного предмета	Общая трудоемкость, ч.	Всего, ч.	Контактная работа, ч			Учебные занятия с применением дистанционных технологий, ч	Самостоятельная работа, ч	Форма аттестации
			Лекции	Лабораторные работы	Практические, семинарские занятия и др. занятия			
2	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Тема 1. “Введение в ООП”	2	0	0	0	0	1	1	-
Тема 2. “Наследование”	4	0	0	0	0	2	2	-
Тема 3. “Перегрузка”	4	0	0	0	0	2	2	-
Тема 4. “Шаблоны и контейнеры”	6	0	0	0	0	3	3	-
Тема 5. “Текстовый квест”	12	0	0	0	0	7	5	-
Промежуточная аттестация	2	0	0	0	0	0	0	Зачёт
Итого:	30	0	0	0	0	15	13	2

3.2. Содержание тем учебного предмета

Тема 1. “Введение в ООП”

- ООП

Тема 2. “Наследование”

- Наследование

Тема 3. “Перегрузка”

- Перегрузка операторов

Тема 4. “Шаблоны и контейнеры”

- Шаблоны
- Контейнеры

Тема 5. “Текстовый квест”

- Текстовый квест

4. Учебно-методическое и информационное обеспечение учебного предмета

№ п/п	Вид и наименование литературы
	Основная
1.	Лебеденко, Л. Ф. Основы программирования на C++ : учебное пособие : [16+] / Л. Ф. Лебеденко, О. И. Моренкова. – 2-е изд., перераб. и доп. – Новосибирск : Сибирский государственный университет телекоммуникаций и информатики, 2021. – 200 с. : ил., табл., схем. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=694769
2.	Информационные технологии и программирование : учебное пособие : [16+] / Е. В. Булгакова, А. Н. Кубанков, М. Д. Хананашвили, Д. С. Дойников. – Москва ; Вологда : Инфра-Инженерия, 2025. – 120 с. : ил., табл. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=725648
3.	Исаева, Г. Н. Языки программирования : практикум по курсу «Языки программирования» : учебное пособие : [16+] / Г. Н. Исаева, Н. В. Логачёва, Ю. В. Стреналюк ; Технологический университет. – Москва : Директ-Медиа, 2024. – 109 с. : ил., табл. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=713440
	Дополнительная
4.	Карчевская, М. П. Основы алгоритмизации инженерных задач : учебное пособие : [16+] / М. П. Карчевская, О. Л. Рамбургер. – Москва ; Вологда : Инфра-Инженерия, 2025. – 244 с. : ил., табл. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=725669
5.	Шапкин, А. С. Задачи с решениями по высшей математике, теории вероятностей, математической статистике, математическому программированию : учебное пособие / А. С. Шапкин, В. А. Шапкин. – 11-е изд., перераб. – Москва : Дашков и К°, 2024. – 402 с. : ил., табл., схем. – (Учебные издания для бакалавров). – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=720215
	Ресурсы информационно-коммуникационной сети «Интернет»
1.	Электронная библиотека на платформе https://lms.mti.moscow/ - https://biblioclub.ru/

5. Учебно-материальная база, необходимая для осуществления образовательного процесса по учебному предмету

Материально-техническое обеспечение учебных предметов включает в себя:

- Персональный компьютер/мобильное устройство (обучающийся обеспечивает себе самостоятельно) с любой операционной системой, позволяющей использовать браузеры и подключаться к сети «Интернет»;
- Рекомендовано установить на персональный компьютер стандартный пакет офисных программ и программ необходимых для выполнения самостоятельной работы (полный список возможно уточнить у куратора программы);
- Обеспечение доступа электронную информационно-образовательную среду Института - lms.mti.moscow;
- Обеспечение доступа в электронную библиотеку biblioclub.ru.

6. Методические рекомендации (указания, материалы) для преподавателей и обучающихся

В процессе изучения данной учебного предмета используются следующие виды учебных занятий и работ: учебные занятия с применением дистанционных образовательных технологий, самостоятельная работа.

6.1 Методические указания для преподавателей при проведении занятий с применением ДОТ.

В процессе занятий с применением ДОТ рекомендуется вести конспект, что позволит впоследствии вспомнить изученный теоретический и практический учебный материал, выполнить самостоятельную работу и подготовиться к итоговой аттестации.

Желательно оставить в рабочих конспектах поля, на которых делать пометки из рекомендованной литературы, дополняющие материал пройденного занятия с применением ДОТ, а также подчеркивающие особую важность тех или иных положений.

На практических занятиях для приобретения умений и навыков целесообразно последовательно выполнять действия за преподавателем, который демонстрирует решение практических заданий, кейсов и т.п.

В завершении занятия с применением ДОТ преподаватель знакомит обучающихся с литературой (основной, дополнительной), с практическими заданиями для самостоятельной работы и даёт рекомендации по их выполнению. Полученную информацию целесообразно кратко и лаконично записывать.

6.2 Методические указания для обучающихся при обучении в виде занятий с применением ДОТ.

В процессе занятий с применением ДОТ рекомендуется вести конспект, что позволит впоследствии вспомнить изученный учебный материал, выполнить самостоятельную работу и подготовиться к промежуточной аттестации по учебному предмету.

Желательно оставить в рабочих конспектах поля, на которых делать пометки из рекомендованной литературы, дополняющие материал прослушанного занятия с применением ДОТ.

Занятия с применением ДОТ имеют логическое завершение, роль которого выполняет заключение. Также в завершении занятия с применением ДОТ преподаватель знакомит обучающихся с литературой (основной, дополнительной), с практическими заданиями для самостоятельной работы и даёт рекомендации по их выполнению. Полученную информацию целесообразно кратко и лаконично записывать.

6.3. Методические указания для обучающихся по выполнению самостоятельной работы

Самостоятельная работа является обязательной для каждого обучающегося, ее объем по учебному предмету определяется учебным планом. Самостоятельная работа предполагает изучение обязательной и дополнительной литературы и выполнение практических заданий, направленных на достижение планируемых результатов по учебному предмету. Практические задания выполняются последовательно с изучением тем учебного предмета и получения знаний в процессе занятий.

6.4. Методические указания для преподавателей по организации самостоятельной работы

Для организации самостоятельной работы обучающихся преподаватель готовит перечень обязательной и дополнительной литературы, а также практические задания, направленные на достижение планируемых результатов по учебному предмету.

Практические задания для самостоятельной работы:

1. **Задание**

Создайте класс Rectangle с двумя закрытыми полями width и height, а также методами для вычисления площади и периметра прямоугольника.

1. Определите класс Rectangle с закрытыми полями width и height.
2. Создайте конструктор с двумя параметрами для инициализации полей width и height.
3. Определите метод area() для вычисления площади прямоугольника.
4. Определите метод perimeter() для вычисления периметра прямоугольника.
5. В функции main() запросите у пользователя длину и ширину прямоугольника, а затем создайте объект класса Rectangle с введёнными значениями ширины и высоты.
6. Вычислите площадь и периметр прямоугольника, используя методы area() и perimeter(), и выведите результаты вычислений на экран
7. В итоге у вас должна получиться следующая программа:

Запустите и протестируйте программу.

2. **Задание**

Создайте базовый класс Животное, в котором будут методы есть(), спать() и издаватьЗвук(). После этого реализуйте классы-наследники Кошка, Собака и Птица, которые будут наследоваться от класса Животное и по-своему переопределять метод издаватьЗвук() для каждого типа животного.

1. Определите базовый класс Animal с методами eat, sleep и makeSound.
2. Реализуйте методы eat и sleep так, чтобы они выводили подходящие текстовые сообщения на экран.
3. Реализуйте метод makeSound как виртуальный, с выводом сообщения на экран.
4. Создайте классы Cat, Dog и Bird, наследуемые от класса Animal.
5. В каждом из производных классов переопределите метод makeSound, чтобы он выводил характерный звук соответствующего животного.
6. В функции main создайте по одному объекту каждого из этих классов.

7. Для каждого созданного объекта вызовите метод makeSound.
8. В результате у вас должна получиться аналогичная по структуре программа:

Запустите программу и проверьте результат.

3. Задание

Создайте вектор целых чисел и заполните его случайными значениями. Затем отсортируйте вектор в порядке возрастания и выведите его содержимое на экран.

1. Подключите необходимые библиотеки: iostream, vector, algorithm, cstdlib и ctime.
2. Инициализируйте генератор случайных чисел текущим временем с помощью функций srand и time.
3. Создайте вектор целых чисел с помощью контейнера vector.
4. Заполните вектор случайными значениями в диапазоне от 0 до 99 с помощью цикла for и функции rand.
5. Отсортируйте вектор в порядке возрастания с помощью алгоритма sort.
6. Выведите содержимое вектора на экран с помощью цикла for и оператора вывода cout.
7. У Вас должна получиться следующая программа:

Запустите программу и проверьте результат.

4. Задание

Создать шаблонную функцию, которая умножает 2 числа. Тип возвращаемого значения должен быть double, функция должна корректно работать для основных числовых типов данных (int, double, float, long long). Протестируйте работы функции.

5. Задание

Создать родительский класс «Товар», а также дочерние классы «Холодильник» и «Чайник», придумать им соответствующие поля, хранящие информацию о типе товара (для дочерних классов), стоимости единицы товара, количества товара на складе. Добавить метод info(), выводящий полную информацию о товаре. Корректно реализовать конструкторы для всех классов. Перегрузить оператор сложения так, чтобы к любому товару можно было прибавлять целое число, в результате чего должно увеличиваться поле, соответствующее количеству товара. Сделать классы шаблонными чтобы стоимость единицы товара могла быть любым подходящим типом данных. Протестируйте работу классов.

6. Задание

Дополнить предыдущее задание. Создать вектор указателей на «Товар», поместить туда экземпляр «Холодильника» и экземпляр «Чайника». Сделать так, чтобы при обходе вектора и вызове метода info() вызывались методы дочерних классов. Добавить к дочерним классам динамическое поле (указатель на int), хранящее информацию об объёме, который товар занимает на складе. В конструкторе выделить память под это поле и задать стартовое значение (очевидно, что холодильник должен

занимать больше места, чем чайник). Корректно реализовать деструкторы для всей иерархии классов. Протестировать корректную работу классов в основной программе. Протестируйте работу методов.

7. Оценочные материалы для промежуточной аттестации обучающихся по учебному предмету

Форма проведения промежуточной аттестации – зачет.

Зачет проводится в виде электронного тестирования.

Общее время, требуемое на выполнение – 2 часа.

Порядок проведения электронного тестирования: проводится дистанционно, предполагает прохождение обучающимися теста, направленного на оценку сформированности планируемых результатов обучения, состоящего из _ вопросов. Каждый правильный ответ оценивается в _ балл_, неправильный оценивается в 0 баллов.

Перечень вопросов, выносимых на электронное тестирование:

1. Что является основной идеей объектно-ориентированного программирования?
    a. инкапсуляция данных и методов в объектах
    b. разделение программы только на функции без данных
    c. использование исключительно глобальных переменных
2. Что такое класс в C++?
    a. шаблон (описание) для создания объектов с общими свойствами и методами
    b. отдельный исполняемый файл программы
    c. тип данных, который существует только во время компиляции и не имеет объектов
3. Что такое объект в терминах ООП?
    a. экземпляр класса с конкретными значениями полей
    b. любая переменная типа int или double
    c. файл исходного кода программы
4. Какой принцип ООП означает сокрытие деталей реализации от пользователя?
    a. инкапсуляция
    b. наследование
    c. полиморфизм
5. Какой принцип ООП позволяет одному интерфейсу иметь множество реализаций?
    a. полиморфизм
    b. инкапсуляция
    c. абстракция памяти
6. Что обычно описывают внутри определения класса?
    a. поля (данные) и методы (функции-члены)
    b. только глобальные константы
    c. только функции main и дополнительные функции
7. Что такое наследование в C++?
    a. создание нового класса на основе существующего с возможностью расширения или изменения поведения
    b. копирование значений переменных между объектами
    c. разделение программы на несколько файлов
8. Как называется класс, от которого наследуются?
    a. базовый (родительский) класс
    b. промежуточный класс

c. локальный класс

9. Какой знак используется для указания наследования в объявлении класса в C++?
    a. двоеточие : после имени класса
    b. стрелка -> после имени класса
    c. знак равенства = после имени класса
10. Какой вид наследования по умолчанию используется в классах (class) в C++?
    a. закрытое (private) наследование
    b. открытое (public) наследование
    c. защищённое (protected) наследование
11. Что позволяет сделать механизм наследования?
    a. повторно использовать код базового класса и добавлять новые возможности
    b. увеличить размер выполняемого файла без пользы
    c. запретить использование методов базового класса
12. Что такое перегрузка операторов в C++?
    a. определение собственного поведения стандартных операторов для пользовательских типов
    b. запрет использования операторов в программе
    c. использование операторов только внутри функции main
13. Какой из операторов можно перегрузить в C++?
    a. -
    b. . (точка доступа к членам)
    c. ?: (тернарный оператор)
14. Как обычно перегрузить бинарный оператор + для класса?
    a. определить функцию-оператор operator+ как метод класса или как внешнюю функцию
    b. изменить стандарт библиотеки c++
    c. переопределить работу оператора в компиляторе
15. Для чего чаще всего перегрудают оператор << для std::ostream?
    a. для удобного вывода объектов пользовательских типов в поток (на экран, в файл)
    b. для изменения работы операции сдвига битов влево
    c. для ускорения работы арифметических выражений
16. Что важно соблюдать при перегрузке операторов?
    a. логичность и ожидаемое поведение, близкое к поведению операторов для встроенных типов
    b. максимальное усложнение кода для защиты от копирования
    c. обязательное изменение приоритета оператора
17. Что такое шаблон (template) в C++?
    a. механизм обобщённого программирования, позволяющий писать код для разных типов
    b. тип специального файла конфигурации компилятора
    c. средство для компоновки нескольких .cpp-файлов
18. Как объявить шаблон функции в C++?
    a. использовать ключевое слово template с параметрами типов перед объявлением функции
    b. добавить слово generic перед типом функции
    c. записать тип auto во всех местах вместо конкретных типов

19. Что такое контейнер в контексте стандартной библиотеки C++ (STL)?
    a. класс, который хранит и управляет набором элементов определённого типа
    b. любая функция стандартной библиотеки
    c. отдельный исполняемый модуль программы
20. Какой контейнер STL представляет собой динамический массив?
    a. std::vector
    b. std::stack
    c. std::queue
21. Какой контейнер STL лучше всего подходит для хранения пар «ключ–значение» с быстрым поиском по ключу?
    a. std::map
    b. std::list
    c. std::stack
22. Что из перечисленного верно для стандартных контейнеров STL?
    a. они работают с итераторами и поддерживают алгоритмы стандартной библиотеки
    b. они могут использоваться только внутри функции main
    c. они не могут хранить объекты пользовательских классов
23. Что такое текстовый квест в программировании?
    a. консольная или текстовая игра, в которой игрок делает выбор через текстовый интерфейс
    b. только игра с 3d-графикой и звуком
    c. тестовая утилита для проверки производительности процессора
24. Как обычно представляют состояние игры в простом текстовом квесте на C++?
    a. с помощью переменных и/или объектов классов, описывающих персонажа, локацию и прогресс
    b. только с помощью глобального массива из чисел
    c. исключительно через макросы препроцессора
25. Что такое l-value?
    a. временный, созданный при выполнении программы, к адресу которого можно обратиться с помощью &
    b. временный объект, созданный при выполнении программы, к адресу которого нельзя обратиться
    c. объект, который имеет конкретный адрес в памяти, к которому можно обратиться

Критерии оценивания промежуточной аттестации:

Перевод полученных баллов по результатам тестирования и отчеты о выполнении практических заданий в отметку производится следующим образом:

90 -100 баллов «отлично»/ «зачтено»;

70 - 89 баллов «хорошо»/ «зачтено»;

50 - 69 баллов «удовлетворительно»/ «зачтено»;

менее 50 баллов «неудовлетворительно»/ «не зачтено».

Образовательная автономная некоммерческая организация

высшего образования

«МОСКОВСКИЙ ТЕХНОЛОГИЧЕСКИЙ ИНСТИТУТ»

Актуализированная версия
утверждена на заседании
Ученого совета
ОАНО ВО «МосТех»
протокол № 07 от 12 февраля 2026 г.

УТВЕРЖДАЮ
Ректор
_____Ю.В. Вепринцева

«12» февраля 2026 г.

.

РАБОЧАЯ ПРОГРАММА УЧЕБНОГО ПРЕДМЕТА «ОПТИМИЗАЦИЯ И ОСНОВЫ АЛГОРИТМИЗАЦИИ»

(наименование программы)

Москва 2026

. Цель изучения учебного предмета

Обучение оптимизации кода на языке C++.

2. Планируемые результаты обучения по учебному предмету

Знать	• Особенности выбранной среды программирования; • Компоненты программно-технических архитектур, существующие приложения и интерфейсы взаимодействия с ними.
Уметь	• Использовать возможности имеющейся технической и/или программной архитектуры для написания программного кода.
Владеть	• Навыком оптимизации программного кода с использованием специализированных программных средств.

3. Содержание учебного предмета

3.1. Распределение учебного времени, выделенного на контактную работу обучающихся с преподавателем, на самостоятельную работу обучающихся и учебные часы с использованием дистанционных образовательных технологий

Общая трудоемкость (объем) учебного предмета составляет 44 академических часа.

	Наименование тем учебного предмета	Общая трудоемкость, ч.	Всего, ч.	Контактная работа, ч			Учебные занятия с применением дистанционных технологий, ч	Самостоятельная работа, ч	Форма аттестации
				Лекции	Лабораторные работы	Практические, семинарские занятия и др. занятия			
1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
1.	Тема 1. “Работа с памятью”	2	0	0	0	0	1	1	-
2.	Тема 2. “Потоки и локи”	4	0	0	0	0	2	2	-
3.	Тема 3. “Основы алгоритмизации”	2	0	0	0	0	1	1	-
4.	Тема 4. “Основы оптимизации”	4	0	0	0	0	2	2	-
5.	Тема 5. “Алгоритмы на списковые структуры данных”	6	0	0	0	0	3	3	-
6.	Тема 6. “Алгоритмы на массивы”	4	0	0	0	0	2	2	-
7.	Тема 7. “Алгоритмы поиска и сортировки”	8	0	0	0	0	4	4	-
8.	Тема 8. “Рекурсивные алгоритмы”	4	0	0	0	0	2	2	-
9.	Тема 9. “Численные алгоритмы”	6	0	0	0	0	4	2	-
10.	Тема 10. “Хэш”	2	0	0	0	0	1	1	-
	Промежуточная аттестация	2	0	0	0	0	0	0	Зачёт
	Итого:	44	0	0	0	0	22	20	2

3.2. Содержание тем учебного предмета

Тема 1. “Работа с памятью”

- Работа с памятью

Тема 2. “Потоки и локи”

- Потоки
- Локи

Тема 3. “Основы алгоритмизации”

- Основы алгоритмизации

Тема 4. “Основы оптимизации”

- Основы оптимизации 1

Тема 5. “Алгоритмы на списковые структуры данных”

- Список
- Стек

Тема 6. “Алгоритмы на массивы”

- Алгоритмы на массивы

Тема 7. “Алгоритмы поиска и сортировки”

- Алгоритмы сортировки
- Алгоритмы поиска

Тема 8. “Рекурсивные алгоритмы”

- Рекурсивные алгоритмы

Тема 9. “Численные алгоритмы”

- Численные алгоритмы

Тема 10. “Хэш”

- Хэш

4. Учебно-методическое и информационное обеспечение учебного предмета

№ п/п	Вид и наименование литературы
	Основная
1.	Лебеденко, Л. Ф. Основы программирования на C++ : учебное пособие : [16+] / Л. Ф. Лебеденко, О. И. Моренкова. – 2-е изд., перераб. и доп. – Новосибирск : Сибирский государственный университет телекоммуникаций и информатики, 2021. – 200 с. : ил., табл., схем. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=694769
2.	Информационные технологии и программирование : учебное пособие : [16+] / Е. В. Булгакова, А. Н. Кубанков, М. Д. Хананашвили, Д. С. Дойников. – Москва ; Вологда : Инфра-Инженерия, 2025. – 120 с. : ил., табл. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=725648
3.	Исаева, Г. Н. Языки программирования : практикум по курсу «Языки программирования» : учебное пособие : [16+] / Г. Н. Исаева, Н. В. Логачёва, Ю. В. Стреналюк ; Технологический университет. – Москва : Директ-Медиа, 2024. – 109 с. : ил., табл. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=713440
	Дополнительная
4.	Карчевская, М. П. Основы алгоритмизации инженерных задач : учебное пособие : [16+] / М. П. Карчевская, О. Л. Рамбургер. – Москва ; Вологда : Инфра-Инженерия, 2025. – 244 с. : ил., табл. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=725669
5.	Шапкин, А. С. Задачи с решениями по высшей математике, теории вероятностей, математической статистике, математическому программированию : учебное пособие / А. С. Шапкин, В. А. Шапкин. – 11-е изд., перераб. – Москва : Дашков и К°, 2024. – 402 с. : ил., табл., схем. – (Учебные издания для бакалавров). – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=720215
	Ресурсы информационно-коммуникационной сети «Интернет»
1.	Электронная библиотека на платформе https://lms.mti.moscow/ - https://biblioclub.ru/

5. Учебно-материальная база, необходимая для осуществления образовательного процесса по учебному предмету

Материально-техническое обеспечение учебных предметов включает в себя:

- Персональный компьютер/мобильное устройство (обучающийся обеспечивает себе самостоятельно) с любой операционной системой, позволяющей использовать браузеры и подключаться к сети «Интернет»;
- Рекомендовано установить на персональный компьютер стандартный пакет офисных программ и программ необходимых для выполнения самостоятельной работы (полный список возможно уточнить у куратора программы);
- Обеспечение доступа электронную информационно-образовательную среду Института - lms.mti.moscow;
- Обеспечение доступа в электронную библиотеку biblioclub.ru.

6. Методические рекомендации (указания, материалы) для преподавателей и обучающихся

В процессе изучения данной учебного предмета используются следующие виды учебных занятий и работ: учебные занятия с применением дистанционных образовательных технологий, самостоятельная работа.

6.1 Методические указания для преподавателей при проведении занятий с

применением ДОТ.

В процессе занятий с применением ДОТ рекомендуется вести конспект, что позволит впоследствии вспомнить изученный теоретический и практический учебный материал, выполнить самостоятельную работу и подготовиться к итоговой аттестации.

Желательно оставить в рабочих конспектах поля, на которых делать пометки из рекомендованной литературы, дополняющие материал пройденного занятия с применением ДОТ, а также подчеркивающие особую важность тех или иных положений.

На практических занятиях для приобретения умений и навыков целесообразно последовательно выполнять действия за преподавателем, который демонстрирует решение практических заданий, кейсов и т.п.

В завершении занятия с применением ДОТ преподаватель знакомит обучающихся с литературой (основной, дополнительной), с практическими заданиями для самостоятельной работы и даёт рекомендации по их выполнению. Полученную информацию целесообразно кратко и лаконично записывать.

6.2 Методические указания для обучающихся при обучении в виде занятий с применением ДОТ.

В процессе занятий с применением ДОТ рекомендуется вести конспект, что позволит впоследствии вспомнить изученный учебный материал, выполнить самостоятельную работу и подготовиться к промежуточной аттестации по учебному предмету.

Желательно оставить в рабочих конспектах поля, на которых делать пометки из рекомендованной литературы, дополняющие материал прослушанного занятия с применением ДОТ.

Занятия с применением ДОТ имеют логическое завершение, роль которого выполняет заключение. Также в завершении занятия с применением ДОТ преподаватель знакомит обучающихся с литературой (основной, дополнительной), с практическими заданиями для самостоятельной работы и даёт рекомендации по их выполнению. Полученную информацию целесообразно кратко и лаконично записывать.

6.3. Методические указания для обучающихся по выполнению самостоятельной работы

Самостоятельная работа является обязательной для каждого обучающегося, ее объем по учебному предмету определяется учебным планом. Самостоятельная работа предполагает изучение обязательной и дополнительной литературы и выполнение практических заданий, направленных на достижение планируемых результатов по учебному предмету. Практические задания выполняются последовательно с изучением тем учебного предмета и получения знаний в процессе занятий.

6.4. Методические указания для преподавателей по организации самостоятельной работы

Для организации самостоятельной работы обучающихся преподаватель готовит перечень обязательной и дополнительной литературы, а также практические задания, направленные на достижение планируемых результатов по учебному предмету.

Практические задания для самостоятельной работы:

1. **Задание**

Написать программу, которая использует умный указатель std::unique_ptr для управления динамически выделенным объектом типа int. Программа должна создать std::unique_ptr, который указывает на динамически выделенное значение int, инициализированное значением 5. Затем программа должна вывести значение, на которое указывает std::unique_ptr, и изменить его на 10. Наконец, программа должна снова вывести значение, на которое указывает std::unique_ptr.

1. Включите заголовочный файл <memory>, который содержит определение std::unique_ptr.
2. В функции main создайте std::unique_ptr, который указывает на динамически выделенное значение int, инициализированное значением 5. Это можно сделать с помощью конструктора std::unique_ptr и оператора new.
3. Выведите значение, на которое указывает std::unique_ptr, используя оператор разыменования (*).
4. Измените значение, на которое указывает std::unique_ptr, на 10, используя оператор разыменования (*).
5. Снова выведите значение, на которое указывает std::unique_ptr, используя оператор разыменования (*).
6. У вас должна получиться следующая программа:

Протестируйте работу программы.

2. **Задание**

Написать программу, которая создает список std::list из 5 элементов и затем удаляет из него элемент с заданным значением.

1. Подключите библиотеки <iostream> и <list>.
2. Создайте пустой список std::list<int> с именем myList. Создайте переменную value типа int для хранения вводимых значений.
3. Выведите на экран приглашение для ввода элементов списка. Считайте 5 значений с помощью цикла for и добавьте их в список myList с помощью метода push_back.
4. Создайте переменную valueToRemove типа int для хранения значения, которое нужно удалить из списка. Выведите на экран приглашение для ввода значения для удаления. Считайте значение и сохраните его в переменной valueToRemove.
5. Используйте метод remove у объекта myList для удаления элемента с заданным значением. Выведите обновлённый список.
6. У вас должна получиться следующая программа:

Протестируйте работу программы.

3. **Задание**

Написать программу, которая использует std::stack для проверки строки на палиндром.

1. Подключите библиотеки <iostream>, <stack> и <string>.

2. Создайте функцию isPalindrome, которая принимает строку в качестве аргумента и возвращает bool-значение.
3. Внутри функции isPalindrome создайте стек символов s. Вычислите длину строки n, пройдите по первой половине строки с помощью цикла for и добавьте символы в стек s с помощью метода push.
4. Определите начало второй половины строки как (n + 1) / 2. Пройдите по второй половине строки с помощью цикла for.
5. На каждой итерации сравните текущий символ строки с верхним элементом стека. Если они не равны, то верните false. Если символы равны, то удалите верхний элемент стека с помощью метода pop. Если цикл завершился успешно, то верните true.
6. В функции main создайте переменную для хранения строки. Выведите на экран приглашение для ввода строки и считайте ее с помощью функции std::getline.
7. Вызовите функцию isPalindrome для проверки строки на палиндромность и выведите результат на экран.
8. В результате у Вас должна получиться следующая программа:

Протестируйте работу программы.

4. Задание

Написать программу, которая использует два потока для вывода чисел от 1 до 10. Один поток должен выводить четные числа, а другой - нечетные. Числа должны выводиться в правильном порядке, переменная-счётчик должна быть глобальной и должна использоваться каждым потоком. Для синхронизации потоков используйте std::mutex. Протестируйте работу программы.

5. Задание

Создайте вручную простой односвязный список. Реализуйте вручную эффективную функцию сортировки этого списка – сортировку вставками (insertions_sort). Протестируйте работу программы.

7. Оценочные материалы для промежуточной аттестации обучающихся по учебному предмету

Форма проведения промежуточной аттестации – зачет.

Зачет проводится в виде электронного тестирования.

Общее время, требуемое на выполнение – 2 часа.

Порядок проведения электронного тестирования: проводится дистанционно, предполагает прохождение обучающимися теста, направленного на оценку сформированности планируемых результатов обучения, состоящего из _ вопросов. Каждый правильный ответ оценивается в _ балл_, неправильный оценивается в 0 баллов.

Перечень вопросов, выносимых на электронное тестирование:

1. Какой оператор в C++ используется для динамического выделения памяти под один объект?
    a. new
    b. malloc
    c. alloc
2. Какой оператор в C++ используется для освобождения памяти, выделенной под массив через new[]?
    a. delete
    b. delete[]
    c. free
3. Что произойдёт при использовании неинициализированного указателя для доступа к памяти?
    a. программа гарантированно завершится без ошибок
    b. возможна неопределённая работа программы (undefined behavior)
    c. указатель автоматически станет нулевым
4. Зачем нужны умные указатели в C++?
    a. Для ускорения работы программы
    b. Для уменьшения размера исполняемого файла программы
    c. Для автоматического освобождения памяти и обеспечения безопасного ее использования
5. Какой заголовочный файл чаще всего подключают для работы с потоками в современном C++?
    a. <thread>
    b. <pthread.h>
    c. <conio.h>
6. Что является первым шагом при решении задачи с точки зрения алгоритмизации?
    a. сразу написать код на C++
    b. понимание и формализация постановки задачи
    c. оптимизация по времени исполнения
7. Какой класс в стандартной библиотеке C++ используется для мьютекса (базового лока)?
    a. std::thread
    b. std::atomic
    c. std::mutex
8. Как работает mutex в C++?
    a. Мьютекс автоматически корректно организует работу нескольких потоков с общим ресурсом, полностью исключая возможность взаимной блокировки
    b. Когда один поток захватывает мьютекс, другие потоки не могут получить доступ к общим данным, пока текущий поток не освободит мьютекс.
    c. Мьютекс помогает нескольким параллельным потокам синхронизировать возвращение результатов выполнений функций в корректном порядке с точки зрения логики, а не просто по мере готовности
9. Какой механизм позволяет безопасно захватывать и автоматически освобождать мьютекс при выходе из области видимости?
    a. std::unique_lock
    b. std::move

c. std::shared_ptr

10. Что в первую очередь обычно оптимизируют в алгоритмах?
    a. внешний вид кода
    b. время выполнения и/или использование памяти
    c. количество комментариев в коде
11. Какой общий подход к оптимизации считается правильным?
    a. сначала измерить (профилировать), потом оптимизировать узкие места
    b. сразу переписать всё на ассемблер
    c. никогда не менять уже работающий код
12. Что означает «асимптотическая сложность алгоритма»?
    a. точное время выполнения алгоритма в секундах
    b. оценка роста времени/памяти при увеличении размера входных данных
    c. количество строк в исходном коде
13. Какая асимптотическая сложность у алгоритма бинарного поиска?
    a. O(log n)
    b. O(n log n)
    c. O(n)
14. Какова основная особенность стека как структуры данных?
    a. доступ к элементам по произвольному индексу
    b. принцип «первым пришёл — первым вышел» (FIFO)
    c. принцип «последним пришёл — первым вышел» (LIFO)
15. Какую операцию обычно выполняют со стеком для добавления нового элемента?
    a. push
    b. enqueue
    c. insert_at
16. Какой доступ к элементам предоставляет обычный массив в C++?
    a. доступ только к первому и последнему элементу
    b. последовательный доступ только по итератору
    c. прямой доступ по индексу за константное время
17. Что произойдёт при выходе за пределы массива (доступ по неверному индексу)?
    a. компилятор автоматически исправит индекс
    b. возможна неопределённая работа программы (undefined behavior)
    c. элемент будет создан автоматически
18. Какой алгоритм сортировки обычно проще всего для понимания и реализации, но неэффективен на больших данных?
    a. быстрая сортировка (quicksort)
    b. пирамидальная сортировка (heapsort)
    c. пузырьковая сортировка (bubble sort)
19. Какой из алгоритмов поиска работает только на отсортированном массиве?
    a. линейный поиск
    b. бинарный поиск
    c. поиск по хэш-таблице
20. Что делает алгоритм линейного поиска?
    a. проверяет элементы последовательно до нахождения искомого или конца
    b. делит массив пополам на каждом шаге
    c. переставляет элементы массива по возрастанию
21. Что является признаком рекурсивного алгоритма?

a. он использует только циклы for и while
b. функция вызывает саму себя с изменёнными параметрами
c. он не может завершиться

22. Какое важное условие нужно для правильной работы рекурсии?
a. отсутствие параметров у функции
b. наличие базового случая (условия завершения)
c. использование только глобальных переменных

23. Что из перечисленного относится к численным алгоритмам?
a. поиск по строке
b. численное решение уравнений
c. сортировка символов

24. Что такое хэш?
a. хэш - это алгоритм компрессии данных, который использует хэш-функцию для уменьшения размера исходных данных
b. хэш - это алгоритм сортировки, который использует хэш-таблицы для ускорения поиска элементов в массиве
c. хэш - это значение фиксированного размера, полученное из исходных данных с помощью хэш-функции

25. Для чего чаще всего используется хэш-таблица?
a. для хранения данных с быстрым доступом по ключу
b. для вывода данных на экран
c. для последовательного чтения файла

Критерии оценивания промежуточной аттестации:

Перевод полученных баллов по результатам тестирования и отчеты о выполнении практических заданий в отметку производится следующим образом:

90 -100 баллов «отлично»/ «зачтено»;

70 - 89 баллов «хорошо»/ «зачтено»;

50 - 69 баллов «удовлетворительно»/ «зачтено»;

менее 50 баллов «неудовлетворительно»/ «не зачтено».

Образовательная автономная некоммерческая организация

высшего образования

«МОСКОВСКИЙ ТЕХНОЛОГИЧЕСКИЙ ИНСТИТУТ»

Актуализированная версия
утверждена на заседании
Ученого совета
ОАНО ВО «МосТех»
протокол № 07 от 12 февраля 2026 г.

УТВЕРЖДАЮ
Ректор
_____Ю.В. Вепринцева

«12» февраля 2026 г.

РАБОЧАЯ ПРОГРАММА УЧЕБНОГО ПРЕДМЕТА

«АЛГОРИТМЫ И ГРАФЫ»

(наименование программы)

Москва 2026

1. Цель изучения учебного предмета

Обучение работе с алгоритмами и графами.

2. Планируемые результаты обучения по учебному предмету

Знать	• Методы и приемы алгоритмизации поставленных задач и работы с графами; • Нотации и программное обеспечение для графического отображения алгоритмов; • Алгоритмы решения типичных задач, области и способы их применения.
Уметь	• Использовать методы и приемы алгоритмизации поставленных задач и работать с графами; • Использовать программное обеспечение для графического отображения алгоритмов; • Применять алгоритмы решения типовых задач в соответствующих областях.
Владеть	• Навыком проверки корректности алгоритмов решения поставленных задач.

3. Содержание учебного предмета

3.1. Распределение учебного времени, выделенного на контактную работу обучающихся с преподавателем, на самостоятельную работу обучающихся и учебные часы с использованием дистанционных образовательных технологий

Общая трудоемкость (объем) учебного предмета составляет 66 академических часов.

Наименование тем учебного предмета	Общая трудоемкость, ч.	Всего, ч.	Контактная работа, ч			Учебные занятия с применением дистанционных технологий, ч	Самостоятельная работа, ч	Форма аттестации
			Лекции	Лабораторные работы	Практические, семинарские занятия и др. занятия			
2	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Тема 1. “Граф”	4	0	0	0	0	2	2	-
Тема 2. “Алгоритмы на древовидные структуры данных”	22	0	0	0	0	12	10	-
Тема 3. “Полезные алгоритмы”	14	0	0	0	0	7	7	-
Тема 4. “Алгоритмы на графы”	10	0	0	0	0	5	5	-
Тема 5. “Строковые алгоритмы”	6	0	0	0	0	3	3	-
Тема 6. “Алгоритмы на собеседованиях”	2	0	0	0	0	1	1	-
Тема 7. “Шашки”	6	0	0	0	0	2	4	-
Промежуточная аттестация	2	0	0	0	0	0	0	Экзамен
Итого:	66	0	0	0	0	32	32	2

3.2. Содержание тем учебного предмета

Тема 1. “Граф”

- Графы

Тема 2. “Алгоритмы на древовидные структуры данных”

- Бинарные деревья
- Дерево отрезков
- Сбалансированные деревья
- Красно-чёрное дерево
- Дерамида

Тема 3. “Полезные алгоритмы”

- Персистентность
- Жадные алгоритмы
- Динамическое программирование
- Теория игр

Тема 4. “Алгоритмы на графы”

- Алгоритмы на графы

Тема 5. “Строковые алгоритмы”

- Алгоритмы на строки

Тема 6. “Алгоритмы на собеседованиях”

- Алгоритмы на собеседованиях

Тема 7. “Шашки”

- Шашки

4. Учебно-методическое и информационное обеспечение учебного предмета

№ п/п	Вид и наименование литературы
	Основная
1.	Лебеденко, Л. Ф. Основы программирования на C++ : учебное пособие : [16+] / Л. Ф. Лебеденко, О. И. Моренкова. – 2-е изд., перераб. и доп. – Новосибирск : Сибирский государственный университет телекоммуникаций и информатики, 2021. – 200 с. : ил., табл., схем. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=694769
2.	Информационные технологии и программирование : учебное пособие : [16+] / Е. В. Булгакова, А. Н. Кубанков, М. Д. Хананашвили, Д. С. Дойников. – Москва ; Вологда : Инфра-Инженерия, 2025. – 120 с. : ил., табл. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=725648
3.	Исаева, Г. Н. Языки программирования : практикум по курсу «Языки программирования» : учебное пособие : [16+] / Г. Н. Исаева, Н. В. Логачёва, Ю. В. Стреналюк ; Технологический университет. – Москва : Директ-Медиа, 2024. – 109 с. : ил., табл. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=713440
	Дополнительная
4.	Карчевская, М. П. Основы алгоритмизации инженерных задач : учебное пособие : [16+] / М. П. Карчевская, О. Л. Рамбургер. – Москва ; Вологда : Инфра-Инженерия, 2025. – 244 с. : ил., табл. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=725669
5.	Шапкин, А. С. Задачи с решениями по высшей математике, теории вероятностей, математической статистике, математическому программированию : учебное пособие / А. С. Шапкин, В. А. Шапкин. – 11-е изд., перераб. – Москва : Дашков и К°, 2024. – 402 с. : ил., табл., схем. – (Учебные издания для бакалавров). – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=720215
	Ресурсы информационно-коммуникационной сети «Интернет»
1.	Электронная библиотека на платформе https://lms.mti.moscow/ - https://biblioclub.ru/

5. Учебно-материальная база, необходимая для осуществления образовательного процесса по учебному предмету

Материально-техническое обеспечение учебных предметов включает в себя:

- Персональный компьютер/мобильное устройство (обучающийся обеспечивает себе самостоятельно) с любой операционной системой, позволяющей использовать браузеры и подключаться к сети «Интернет»;
- Рекомендовано установить на персональный компьютер стандартный пакет офисных программ и программ необходимых для выполнения самостоятельной работы (полный список возможно уточнить у куратора программы);
- Обеспечение доступа электронную информационно-образовательную среду Института - lms.mti.moscow;
- Обеспечение доступа в электронную библиотеку biblioclub.ru.

6. Методические рекомендации (указания, материалы) для преподавателей и обучающихся

В процессе изучения данной учебного предмета используются следующие виды учебных занятий и работ: учебные занятия с применением дистанционных образовательных технологий, самостоятельная работа.

6.1 Методические указания для преподавателей при проведении занятий с применением ДОТ.

В процессе занятий с применением ДОТ рекомендуется вести конспект, что позволит впоследствии вспомнить изученный теоретический и практический учебный материал, выполнить самостоятельную работу и подготовиться к итоговой аттестации.

Желательно оставить в рабочих конспектах поля, на которых делать пометки из рекомендованной литературы, дополняющие материал пройденного занятия с применением ДОТ, а также подчеркивающие особую важность тех или иных положений.

На практических занятиях для приобретения умений и навыков целесообразно последовательно выполнять действия за преподавателем, который демонстрирует решение практических заданий, кейсов и т.п.

В завершении занятия с применением ДОТ преподаватель знакомит обучающихся с литературой (основной, дополнительной), с практическими заданиями для самостоятельной работы и даёт рекомендации по их выполнению. Полученную информацию целесообразно кратко и лаконично записывать.

6.2 Методические указания для обучающихся при обучении в виде занятий с применением ДОТ.

В процессе занятий с применением ДОТ рекомендуется вести конспект, что позволит впоследствии вспомнить изученный учебный материал, выполнить самостоятельную работу и подготовиться к промежуточной аттестации по учебному предмету.

Желательно оставить в рабочих конспектах поля, на которых делать пометки из рекомендованной литературы, дополняющие материал прослушанного занятия с применением ДОТ.

Занятия с применением ДОТ имеют логическое завершение, роль которого выполняет заключение. Также в завершении занятия с применением ДОТ преподаватель знакомит обучающихся с литературой (основной, дополнительной), с практическими заданиями для самостоятельной работы и даёт рекомендации по их выполнению. Полученную информацию целесообразно кратко и лаконично записывать.

6.3. Методические указания для обучающихся по выполнению самостоятельной работы

Самостоятельная работа является обязательной для каждого обучающегося, ее объем по учебному предмету определяется учебным планом. Самостоятельная работа предполагает изучение обязательной и дополнительной литературы и выполнение практических заданий, направленных на достижение планируемых результатов по учебному предмету. Практические задания выполняются последовательно с изучением тем учебного предмета и получения знаний в процессе занятий.

6.4. Методические указания для преподавателей по организации самостоятельной работы

Для организации самостоятельной работы обучающихся преподаватель готовит перечень обязательной и дополнительной литературы, а также практические задания, направленные на достижение планируемых результатов по учебному предмету.

Практические задания для самостоятельной работы:

1. **Задание**

Дан взвешенный неориентированный граф. Пользователь вводит его в консоль в виде списка рёбер. Напишите программу, которая считает этот граф и сохранит его сперва в виде списка смежности, а потом в виде матрицы смежности.

1. Считайте количество вершин и ребер графа:

```
int n, m; // количество вершин и ребер
cout << "Enter the number of vertices and edges: ";
cin >> n >> m;
```

2. Создайте структуры данных для хранения графа: список смежности и матрицу смежности:

```
vector<vector<pair<int, int>>> adj_list(n + 1); // список смежности
vector<vector<int>> adj_matrix(n + 1, vector<int>(n + 1)); // матрица смежности
```

3. Считайте ребра графа и добавьте их в список смежности и матрицу смежности:

```
cout << "Enter the edges (vertex1 vertex2 weight):" << endl;
for (int i = 0; i < m; i++) {
   int v1, v2, w;
   cin >> v1 >> v2 >> w;
   // добавляем ребро в список смежности
   adj_list[v1].push_back({ v2,w });
   adj_list[v2].push_back({ v1,w });
   // добавляем ребро в матрицу смежности
   adj_matrix[v1][v2] = w;
   adj_matrix[v2][v1] = w;
}
```

4. Выведите список смежности на экран:

```
cout << "Adjacency list:" << endl;
for (int i = 1; i <= n; i++) {
   cout << i << ": ";
   for (auto j : adj_list[i]) {
      cout << "(" << j.first << ", " << j.second << ") ";
   }
   cout << endl;
}
```

5. Выведите матрицу смежности на экран:

```
cout << "Adjacency matrix:" << endl;
for (int i = 1; i <= n; i++) {
   for (int j = 1; j <= n; j++) {
      cout << adj_matrix[i][j] << " ";
   }
   cout << endl;
}
```

6. У вас должна получиться следующая программа:

Протестируйте программу.

2. **Задание**

Напишите свою реализацию бинарного дерева поиска. Должны быть реализованы методы добавления элементов, удаления элементов, поиска элемента

1. Определите класс, пропишите поля и сигнатуры методов. Поскольку наши методы будут напрямую работать с узлами дерева, то они должны быть закрытыми, то есть приватными. Поэтому нужно будет реализовать публичные методы, которые вызывают соответствующие приватные.

```
#include <iostream>

class BinarySearchTree {
public:
   // Конструктор для создания пустого дерева
   BinarySearchTree() : root(nullptr) { }

   // Публичный метод для вставки нового значения в дерево
   void insert(int value) {
      root = insert(root, value);
   }

   // Публичный метод для удаления значения из дерева
   void remove(int value) {
      root = remove(root, value);
   }

   // Публичный метод для поиска значения в дереве
   bool search(int value) {
      return search(root, value) != nullptr;
   }

   // Публичный метод для печати дерева в консоль
   void print() {
      inorder();
    std::cout << std::endl;
   }

private:
   // Определение структуры узла дерева
   struct Node {
   };

   Node* root; // Корень дерева

   // Приватный метод для вставки нового значения в дерево
   Node* insert(Node* root, int value) {
      // ...
```

```
    }

    // Приватный метод для удаления значения из дерева
    Node* remove(Node* root, int value) {
        // ...
    }

    // Приватный метод для поиска значения в дереве
    Node* search(Node* root, int value) {
        // ...
    }

    // Приватный метод для печати узлов дерева в порядке возрастания
    void inorder() {
        // ...
    }
};
```

2. Определите структуру узла дерева, которая будет содержать значение узла и указатели на левого и правого потомка. Создайте функцию для создания нового узла с заданным значением.

```
// Определение структуры узла дерева
struct Node {
    int value; // Значение узла
    Node* left; // Указатель на левого потомка
    Node* right; // Указатель на правого потомка

    // Конструктор для создания нового узла с заданным значением
    Node(int value) : value(value), left(nullptr), right(nullptr) {}
};
```

3. Создайте функцию для вставки нового узла в дерево. Эта функция должна принимать корень дерева и значение нового узла в качестве аргументов и возвращать новый корень дерева.

```
// Приватный метод для вставки нового значения в дерево
Node* insert(Node* root, int value) {
    // Если дерево пустое, создаем новый корень
    if (root == nullptr) {
        return new Node(value);
    }

    // Если значение меньше значения корня, вставляем его в левое поддерево
    if (value < root->value) {
        root->left = insert(root->left, value);
    }
    // Иначе вставляем его в правое поддерево
    else {
        root->right = insert(root->right, value);
```

```
    }

    // Возвращаем новый корень дерева
    return root;
}
```

4. Создайте функцию для поиска значения в дереве. Эта функция должна принимать корень дерева и значение для поиска в качестве аргументов и возвращать указатель на узел с этим значением или nullptr, если значение не найдено.

```
// Приватный метод для поиска значения в дереве
Node* search(Node* root, int value) {
    // Если дерево пустое или значение найдено, возвращаем корень
    if (root == nullptr || root->value == value) {
        return root;
    }

    // Если значение меньше значения корня, ищем его в левом поддереве
    if (value < root->value) {
        return search(root->left, value);
    }
    // Иначе ищем его в правом поддереве
    else {
        return search(root->right, value);
    }
}
```

5. Создайте функцию для удаления узла из дерева. Эта функция должна принимать корень дерева и значение для удаления в качестве аргументов и возвращать новый корень дерева. Так как нам нужно сохранить свойства бинарного дерева, то для удаления элементов с обоими детьми нам понадобится вспомогательный метод, который ищет преемника. В качестве преемника будем выбирать наименьший элемент левого поддерева.

```
// Приватный метод для нахождения узла с минимальным значением в дереве
Node* findMin(Node* root) {
    while (root->left != nullptr) {
        root = root->left;
    }
    return root;
}



// Приватный метод для удаления значения из дерева
Node* remove(Node* root, int value) {
    // Если дерево пустое, возвращаем nullptr
    if (root == nullptr) {
        return nullptr;
```

```
    }

    // Если значение меньше значения корня, удаляем его из левого поддерева
    if (value < root->value) {
        root->left = remove(root->left, value);
    }
    // Если значение больше значения корня, удаляем его из правого поддерева
    else if (value > root->value) {
        root->right = remove(root->right, value);
    }
    // Иначе удаляем корень
    else {
        // Если у корня нет потомков или только один потомок
        if (root->left == nullptr) {
            Node* temp = root->right;
            delete root;
            return temp;
        }
        else if (root->right == nullptr) {
            Node* temp = root->left;
            delete root;
            return temp;
        }
        // Если у корня есть оба потомка
        else {
            // Находим узел с минимальным значением в правом поддереве
            Node* temp = findMin(root->right);
            // Копируем значение этого узла в корень
            root->value = temp->value;
            // Удаляем этот узел из правого поддерева
            root->right = remove(root->right, temp->value);
        }
    }

    // Возвращаем новый корень дерева
    return root;
}
```

6. Создайте функцию для печати узлов дерева в порядке возрастания. Так как эта функция должна осуществлять обход дерева в глубину, что делается рекурсивно, то нам понадобится функция-помощник.

```
// Приватный метод для печати узлов дерева в порядке возрастания
void inorder() {
    inorderHelper(root);
}

// Приватный рекурсивный метод-помощник для печати
```

```
void inorderHelper(Node* root) {
    // Если дерево не пустое
    if (root != nullptr) {
        // Обходим левое поддерево
        inorderHelper(root->left);
        // Печатаем значение корня
        std::cout << root->value << ' ';
        // Обходим правое поддерево
        inorderHelper(root->right);
    }
}
```

7. Добавьте тесты в функцию main(), в итоге у Вас должен получиться следующий код:

Протестируйте программу на своих тестах.

3. **Задание**

Дано некоторое количество монет различных номиналов и сумма, которую нужно набрать этими монетами. Найдите минимальное количество монет, которыми можно набрать эту сумму. Данную задачу можно решить с помощью жадного алгоритма.

4. **Задание**

Дано некоторое количество занятий с определенными временами начала и окончания. Найдите максимальное количество занятий, которые можно посетить, если можно посещать только одно занятие в каждый момент времени. Данную задачу можно решить с помощью жадного алгоритма.

5. **Задание**

Дан массив чисел. Найти максимальное произведение подпоследовательности из k элементов. Данную задачу можно решить, используя динамическое программирование.

7. Оценочные материалы для промежуточной аттестации обучающихся по учебному предмету

Форма проведения промежуточной аттестации – экзамен.

Экзамен проводится в виде электронного тестирования.

Общее время, требуемое на выполнение – 2 часа.

Порядок проведения электронного тестирования: проводится дистанционно, предполагает прохождение обучающимися теста, направленного на оценку сформированности планируемых результатов обучения, состоящего из 25 вопросов. Каждый правильный ответ оценивается в 4 балла, неправильный оценивается в 0 баллов.

Перечень вопросов, выносимых на электронное тестирование:

1. Что такое граф в теории графов?
    a. невзвешенное множество вершин без рёбер
    b. структура из вершин и рёбер, соединяющих пары вершин
    c. упорядоченный массив чисел
2. Чем ориентированный граф отличается от неориентированного?
    a. в ориентированном графе рёбра имеют направление
    b. в ориентированном графе нет петель
    c. в неориентированном графе нет вершин степени 1
3. Что называется степенью вершины в неориентированном графе?
    a. количество всех вершин графа
    b. количество рёбер, инцидентных этой вершине
    c. номер вершины в списке вершин
4. Какой из способов представления графа обычно эффективен для разреженных графов?
    a. матрица смежности
    b. список смежности
    c. матрица расстояний
5. Что такое бинарное дерево?
    a. дерево, у каждой вершины которого не более двух детей
    b. дерево, у которого все листья на одном уровне
    c. дерево, у которого у каждой вершины ровно два ребёнка
6. Что такое двоичное дерево поиска (BST)?
    a. бинарное дерево без повторяющихся значений
    b. бинарное дерево, в котором левое поддерево содержит элементы меньше ключа, а правое — больше
    c. бинарное дерево, в котором ключи хранятся только в листьях
7. Какова средняя асимптотическая сложность поиска элемента в сбалансированном бинарном дереве поиска?
    a. $O(\log n)$
    b. $O(n)$
    c. $O(1)$
8. Для чего используют дерево отрезков?
    a. для хранения строк и поиска подстрок
    b. для обработки запросов на диапазоны (сумма, минимум и т.п.) в массиве
    c. для сортировки массива пузырьком

9. Какая операция обычно поддерживается деревом отрезков?
    a. поиск подстроки в строке
    b. поиск кратчайшего пути в графе
    c. обновление значения на отрезке
10. В чём ключевая идея сбалансированных деревьев?
    a. поддерживать высоту дерева близкой к логарифмической
    b. хранить все элементы только в листьях
    c. не допускать дубликатов в узлах
11. Для чего в красно-чёрном дереве используется окраска вершин?
    a. для различения листьев и корня
    b. для упрощения вывода дерева на экран
    c. для поддержания инвариантов, гарантирующих балансировку
12. Какова типичная асимптотическая сложность вставки в красно-чёрное дерево?
    a. $O(\log n)$
    b. $O(n^2)$
    c. $O(1)$
13. Что такое пирамида (дерамида, куча) в контексте структур данных?
    a. дерево, у которого все листья имеют одинаковую глубину
    b. полное бинарное дерево, удовлетворяющее свойству кучи
    c. произвольный граф с циклами
14. Какая структура данных чаще всего используется при реализации приоритетной очереди?
    a. очередь FIFO
    b. куча (heap)
    c. стек
15. Что означает персистентность структуры данных?
    a. структура данных может работать без памяти
    b. структура данных не изменяется, а при изменениях создаются новые версии с общим прошлым
    c. структура данных хранит только текущее состояние без истории
16. В чём суть жадных алгоритмов?
    a. они всегда перебирают все варианты
    b. они в каждый момент выбирают локально наилучшее решение в надежде на глобальный оптимум
    c. они используют рекурсию и мемоизацию
17. Что является характерной чертой динамического программирования?
    a. использование случайного выбора
    b. разбиение задачи на перекрывающиеся подзадачи и сохранение результатов
    c. отсутствие использования памяти
18. Для чего в задачах используется мемоизация?
    a. для ускорения ввода-вывода
    b. для избежания повторных вычислений одних и тех же подзадач
    c. для проверки корректности кода
19. Что изучает теория игр в контексте алгоритмов?
    a. стратегии взаимодействия нескольких рациональных игроков
    b. оптимальные методы сортировки

c. построение минимальных остовных деревьев

20. Какой тип задач чаще всего моделируется в теории игр при решении задач на алгоритмы?
    a. задачи с одним исполнителем и детерминированным исходом
    b. задачи о случайном блуждании по графу
    c. игры с двумя игроками и полной информацией (например, крестики-нолики)
21. Какой алгоритм относится к базовым алгоритмам на графах?
    a. быстрая сортировка
    b. поиск в глубину (DFS)
    c. алгоритм Кнута–Морриса–Пратта
22. Что позволяет найти алгоритм Дейкстры?
    a. минимальное остовное дерево
    b. кратчайшие пути от одной вершины до всех остальных в графе с неотрицательными весами
    c. все циклы в неориентированном графе
23. Какой алгоритм относится к строковым алгоритмам поиска подстроки?
    a. алгоритм Беллмана–Форда
    b. алгоритм Кнута–Морриса–Пратта (КМП)
    c. алгоритм Флойда–Уоршелла
24. Для чего используется Z-функция или префикс-функция в строковых алгоритмах?
    a. для нахождения кратчайшего пути в строке
    b. для вычисления хеш-значений чисел
    c. для эффективного поиска образца в тексте и анализа повторяющихся подстрок
25. Задачи какого типа чаще всего встречаются на алгоритмических собеседованиях по C++?
    a. преимущественно вопросы по синтаксису без кода
    b. задачи на структуры данных, строки, графы, динамическое программирование и жадные алгоритмы
    c. только задачи на теорию чисел и сложные интегралы

Критерии оценивания промежуточной аттестации:

Перевод полученных баллов по результатам тестирования и отчеты о выполнении практических заданий в отметку производится следующим образом:

90 -100 баллов «отлично»/ «зачтено»;

70 - 89 баллов «хорошо»/ «зачтено»;

50 - 69 баллов «удовлетворительно»/ «зачтено»;

менее 50 баллов «неудовлетворительно»/ «не зачтено».