

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
 «ДОНСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

**(ДГТУ)**

**МЕТОДИЧЕСКИЕ УКАЗАНИЯ И ЗАДАНИЯ**

**ДЛЯ ВЫПОЛНЕНИЯ КОНТРОЛЬНОЙ РАБОТЫ ПО КУРСУ**

«Высокоуровневые методы информатики и программирования»

**(для студентов заочной формы обучения)**

**Ростов – на – Дону**

**2025**

# Кафедра «Информационные технологии»

**Составитель:**

**Рашидова Е.В.**

Методические указания содержат варианты контрольных работ по курсу «Высокоуровневые методы информатики и программирования» для студентов направления 09.03.03 заочной формы обучения и рекомендации по их выполнению.

**Методические рекомендации для студентов  
 по изучению дисциплины «Высокоуровневые методы информатики и программирования»**

Для оптимальной организации процесса изучения дисциплины студентам предлагаются следующие методические рекомендации.

**I.** **Выбор вопросов, входящих в контрольную работу**.

Каждая контрольная работа состоит из двух вопросов. Вопросы**,** на которые нужно ответить, определяются из таблицы 1 по двум последним цифрам шифра зачетной книжки. Так как в таблице имеется только 49 вариантов, то, в случае если последние две цифры образуют число, большее 49, то для выбора номера варианта от него нужно отнять50.

Например, номер зачетной книжки 1853797. Последние две цифры образуют число 97. Находим номер варианта: 97 – 50 = 47.

###### Таблица 1.

|  |  |  |  |
| --- | --- | --- | --- |
| Шифр | Вопросы | Шифр | Вопросы |
| 1 семестр | 1 семестр | |
| **00** | 1, 42 | **25** | 26, 67 | |
| **01** | 2, 43 | **26** | 27, 68 | |
| **02** | 3, 44 | **27** | 28, 69 | |
| **03** | 4. 45 | **28** | 29, 70 | |
| **04** | 5, 46 | **29** | 30. 71 | |
| **05** | 6, 47 | **30** | 31, 72 | |
| **06** | 7. 48 | **31** | 32, 73 | |
| **07** | 8, 49 | **32** | 33. 74 | |
| **08** | 9. 50 | **33** | 34, 75 | |
| **09** | 10,51 | **34** | 35. 76 | |
| **10** | 11, 52 | **35** | 36,77 | |
| **11** | 12, 53 | **36** | 37, 78 | |
| **12** | 13, 54 | **37** | 38, 79 | |
| **13** | 14, 55 | **38** | 39, 80 | |
| **14** | 15, 56 | **39** | 40, 81 | |
| **15** | 16, 57 | **40** | 41, 82 | |
| **16** | 17. 58 | **41** | 1, 83 | |
| **17** | 18, 59 | **42** | 2. 84 | |
| **18** | 19, 60 | **43** | 3, 42 | |
| **19** | 20. 61 | **44** | 4, 43 | |
| **20** | 21, 62 | **45** | 5. 44 | |
| **21** | 22. 63 | **46** | 6, 45 | |
| **22** | 23,64 | **47** | 7. 46 | |
| **23** | 24, 65 | **48** | 8,47 | |
| **24** | 25, 66 | **49** | 9, 48 | |

**II.** **Требования к** с**одержательной части контрольной работы.**

Контрольные работы по курсу теория информации носят характер письменного реферата. Формулировка вопросов определяет только минимальное содержание ответа на каждый вопрос. Подробные требования к широте охвата темы определяются на установочных лекциях преподавателем.

Несмотря на большое разнообразие изучаемых в курсе тем, средний объем текстовой части ответа на один вопрос должен соответствовать не менее, чем одной-двум страницам машинописного текста.

Ответы должны содержать сведения по существу вопросов. Теоретические вопросы должны поясняться соответствующими примерами. При необходимости должны быть приведены таблицы, графики, схемы и другие иллюстративные материалы с подробными пояснениями.

**III.** **Оформление контрольной работы**.

Первым пунктом контрольной работы должно быть определение номера варианта, по которому будет выполняться контрольная работа. В результате должны быть указаны номера блоков и номера заданий в этих блоках, взятые из таблицы.

Ответ на каждый вопрос должен начинаться с формулировки этого вопроса и заканчиваться списком литературы, использованной при ответе на данный вопрос. Для каждого литературного источника необходимо дополнительно указать использованные разделы и страницы.

После проверки контрольной работы преподавателем с каждым студентом проводится собеседование (защита контрольной работы) по охваченным в ней темам.

ВОПРОСЫ ПО КУРСУ

«Высокоуровневые методы информатики и программирования»

1. Динамическое выделение и перераспределение памяти
2. Операторы управления динамической памятью (в С++)
3. Структуры данных. Линейные списки.
4. Краткий обзор С++. Объектно-ориентированное программирование (ООП). Понятие объект.
5. Фундаментальные понятия ООП. Класс
6. Полиморфизм
7. Наследование
8. Консольный ввод-вывод в С++
9. Перегрузка функций
10. Введение в классы. Конструкторы и деструкторы. Конструкторы с параметрами
11. Указатели на объект
12. Присваивание объектов
13. Дружественные функции
14. Передача объектов функции
15. Объекты в качестве возвращаемого значения функции
16. Ссылки
17. Преимущества параметров-ссылок по сравнению с параметрами-указателями
18. Ссылка в качестве возвращаемого значения
19. Перегрузка конструкторов
20. Создание и использование конструктора копий
21. Перегрузка операторов
22. Перегрузка бинарных операторов
23. Перегрузка операторов отношения и логических операторов
24. Перегрузка унарных операторов
25. Дружественные операторы функции
26. Особенности использования оператора присваивания
27. Перегрузка оператора индекса массива
28. Наследование. Управление доступом к базовому классу.
29. Защищенные члены класса
30. Конструкторы, деструкторы и наследование. Синтаксис передачи аргументов из производного в базовый класс
31. Множественное наследование
32. Виртуальные базовые классы
33. Указатели на производные типы
34. Виртуальные функции. Отличие переопределения виртуальных функций в производных классах
35. Вызов виртуальной функции ссылкой на объект базового класса. Атрибут virtual наследуется
36. Потоки и файлы
37. Текстовые потоки
38. Бинарные потоки
39. Файлы
40. Указатель файла
41. Открытие файла
42. Закрытие файла
43. Запись символа
44. Чтение символа
45. Примеры использования функций: fopen(), fgetc(), fputc(), fclose()
46. Библиотека STL. Основные понятия и классы. Методы работы в STL
47. История создания кроссплатформенных систем программирования.
48. Концепции кроссплатформенного программного обеспечения
49. Характеристики существующих платформ разработки и области их применения
50. Современное состояние и принципиальные возможности языков программирования Java, C#, C++, python;
51. Объектно-ориентированное программирование как доминирующая парадигма программирования.
52. Введение в язык Java. История. Основные направления Java.
53. Синтаксис языка Java. Создание простой программы. Основы GUI. События в Java.
54. JDK и JRE
55. Особенности архитектуры .NET. Среды разработки для Net.
56. Методы и средства разработки приложений с использованием Visual Studio.NET.
57. Синтаксис языка C#.
58. Типы данных в Java, C#, C++, Python.
59. Абстракции и объекты.
60. Абстрактные классы.
61. Интерфейсы.
62. Управление доступом.
63. Инкапуляция.
64. Наследование и полиморфизм.
65. Коллекции объектов.
66. Обработка ошибок и исключения.
67. Внутренние и анонимные (безымянные) внутренние классы.
68. Система ввода-вывода.69. События и их обработка
69. Концепция Model-View-Controller.
70. Портируемость Python.
71. Ссылочные и атомарные объекты.
72. Объектно-ориентированное и функциональное программирование в Python.
73. Модули в языке Python. Модули, написанные на «чистом» Python, и модули расширения (extension modules), написанные на других языках программирования.
74. Стандартная библиотека Python.
75. Взаимодействие с API
76. Платформа Microsoft .NET Framework 4.0. CLR, FCL, JIT compiler. Компиляция и выполнение программы в среде CLR.
77. Встроенные типы языка C#. Создание пользовательских типов. Значимые и ссылочные типы.
78. Состав языка C#. Правила именования идентификаторов рекомендации. Объектно-ориентированное программирование в C#.
79. Создание перечислимых типов. Сравнение ссылочных и значимых типов. Передача ссылочных и значимых типов в подпрограммы. Особенности.
80. Виды массивов в С#. Инициализация массивов. Выделение памяти. Класс Array.
81. Способы представления строк в C#. Преимущества и недостатки каждого способа.
82. Массивы как коллекции. Использование цикла foreach для работы с массивами
83. Перегрузка методов в языке C#. Рекурсивные методы. Методы с переменным числом аргументов.39. Индексаторы. Определение индексаторов для своего класса. Перегрузка других (унарных и бинарных) операций. Примеры перегрузки.
84. Концепция деструкторов в C#. Перегрузка деструкторов. Конструкторы и перегрузка конструкторов. Статические и динамические конструкторы.