**КОНТРОЛЬНЫЕ ВОПРОСЫ К ЭКЗАМЕНУ**

**ПО ДИСЦИПЛИНЕ**

**«ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ»**

**2-Й КУРС, 3-Й СЕМЕСТР ОБУЧЕНИЯ**

**1–40 05 01-02 «Информационные системы и технологии (в экономике)»**

**1–40 05 01-08 «Информационные системы и технологии (в логистике)»**

**1–40 05 01-12 «Информационные системы и технологии (в игровой индустрии)»**

1. Парадигмы программирования, основные направления в программировании.
2. Возникновение объектно-ориентированного программирования.
3. Базовые принципы и конструкции ООП.
4. Особенности процедурного и объектно-ориентированного программирования.
5. Основные понятия и положения объектной модели. Её преимущества и недостатки.
6. Абстрагирование. Модульность. Иерархия. Типизация.
7. Организация ввода/вывода данных в языке С++. Традиционный ввод-вывод ев языке С.
8. Использование манипуляторов для ввода/вывода в языке С++.
9. Взаимодействие потокового и традиционного ввода-вывода.
10. Максимизация эффективности использования памяти компьютера. Особенности и необходимость использования динамически выделяемой памяти.
11. Динамическое распределение памяти с использованием операторов языка С++ new и delete. Особенности выделения и освобождения динамической памяти.
12. Использование new и delete для реализации массивов (одномерных и двумерных) объектов. Инициализация динамически выделенной памяти.
13. Базовые конструкции объектно-ориентированных программ: классы и объекты.
14. Концепция, определение, реализация класса. Принцип инкапсуляции. Атрибуты доступа.
15. Инициализация и разрушение объекта. Способы доступа к содержимому класса. Элементы и методы класса.
16. Конструкторы и деструкторы класса. Конструктор по умолчанию.
17. Конструктор копирования. Конструктор *explicit*. Параметризованные конструкторы.
18. Сравнение классов и структур. Сравнение классов и объединений.
19. Организация внешнего доступа к локальным компонентам класса. Встраиваемые функции класса.
20. Использование встраиваемых методов в определении класса. Массивы объектов класса.
21. Инициализация массивов объектов. Указатель this. Указатели на объекты.
22. Динамическая и статическая инициализация переменных и объектов.
23. Дружественные (*friend*) функции класса. Перегрузка функций и методов. Особенности перегрузки операторов как метода класса и как дружественной функции.
24. Перегрузка конструкторов класса. Применение динамической инициализации к конструкторам.
25. Присваивание объектов. Передача объектов в качестве аргументов функции. Особенности использования конструкторов, деструкторов при передаче объектов в качестве параметров.
26. Потенциальные проблемы при передаче объектов в качестве параметров. Возврат объектов функциями. Потенциальная проблема при возврате объектов функциями.
27. Создание, особенности использование конструктора копии при передаче объекта в качестве параметра функции, возврате объекта из функции и инициализации объектов.
28. Особенности использования конструктора копии в других языках программирования. Локальные static-переменные. Глобальные static-переменные.
29. Особенности выделения памяти при работе с классами и объектами. Статические (static) компоненты класса. Применение статических полей класса. Раздельное объявление и определение статических полей класса. Применение статических методов класса.
30. Использование ключевого слова *const* при создании переменной и объекта класса. Константные компоненты класса. Константные аргументы методов класса. Вложенные классы.
31. Перегрузка операторов для пользовательских типов данных. Перегрузка унарных операторов. Перегрузка бинарных операторов. Особенности использования ключевого слова operator.
32. Аргументы перегружаемого оператора. Значения, возвращаемые перегружаемым оператором. Временные безымянные объекты.
33. Перегрузка постфиксных операторов. Перегрузка оператора ().
34. Перегрузка арифметических операторов. Перегрузка операций ++ и – –
35. Перегрузка оператора []. Перегрузка оператора ->.
36. Перегрузка оператора \*. Перегрузка оператора присваивания
37. Особенности преобразования типов данных. Явные преобразования. Преобразования типов, определенных в программе. Особенности преобразования основных типов в основные типы.
38. Особенности преобразования объектов в основные типы данных и наоборот. Преобразование объектов производных классов в объекты базового класса.
39. Особенности использования указателей и ссылок. Ссылки. Параметры ссылки. Статические и динамические объекты.
40. Наследование, базовый и производный классы. Простое наследование. Открытое, защищенное и закрытое.
41. Определение производного класса, доступ к содержимому базового класса. Использование конструкторов базового класса, использование методов базового класса.
42. Спецификатор доступа protected, недостатки использования спецификатора protected.
43. Неизменность базового класса. Конструкторы производного класса. Переопределение методов базового класса.
44. Использование операции разрешения при переопределении методов. Конструкторы и методы классов при наследовании.
45. Иерархия классов. Абстрактные типы данных. Абстрактный базовый класс.
46. Общее, частное, защищенное наследование. Комбинации атрибутов доступа.
47. Выбор атрибута доступа при наследовании. Уровни наследования. Множественное наследование.
48. Методы классов и множественное наследование. Неопределенности при множественном наследовании.
49. Конструкторы при множественном наследовании. Конструкторы без аргументов и конструкторы с аргументами при множественном наследовании.
50. Связь включения: классы в классах. Композиция и наследование. Композиция: сложное включение. Роль наследования при разработке программ.
51. Указатели и ссылки на производные типы. Понятие и сравнение раннего связывания с поздним.
52. Множественное наследование и виртуальное наследование базовых классов. Использование виртуального механизма для реализации принципа полиморфизма.
53. Виртуальные методы класса и механизм их использования. Чисто виртуальные функции и абстрактные базовые классы.
54. Виртуальный деструктор. Абстрактные классы их назначение и свойства.
55. Наследование виртуальных методов. Необходимость применения виртуальных функции.
56. Полиморфизм и пуризм. Понятие параметризированных функций.
57. Параметризированные функции с двумя и более обобщенными типами. Перегрузка параметризированной функции.
58. Использование стандартных параметров в параметризированных функциях и методах. Особенности использования параметризированных функций и методов.
59. Параметризированные классы, методы, их свойства. Явно задаваемые специализации параметризированных классов.
60. Совместное использование параметризации и принципов наследования.
61. Организация внешнего доступа к компонентам параметризированных классов. Параметризированные классы и статические элементы.
62. Использование в параметризированных классах аргументов, не являющихся типами. Использование в параметризированных классах аргументов по умолчанию.
63. Шаблонные (параметризированные) классы и дружественные функции класса. Спецификатор register.
64. Пространство имен. Пространство имен как объявление. Пространство имен как директива. Определение пространства имен. Доступ к элементам пространств имен. Неоднократное определение пространств имен. Неименованные пространства имен
65. Многофайловые программы как способ структуризации разрабатываемого программного обеспечения. Структура многофайловых приложений на C/C++. Использование ключевого слова *extern*, *static* для организации работы многофайлового приложения.
66. Компоновщик, компилятор. Этапы создания приложения на C/C++. Переименование типов с помощью оператора typedef
67. Причины использования многофайловых программ. Библиотеки классов. Реализация библиотеки классов
68. Создание многофайловой программы. Директории в многофайловых программах
69. Проекты в IDE Visual Studio. Взаимодействие исходных файлов. Заголовочные файлы. Межфайловые переменные, межфайловые функции, межфайловые классы. Ошибка повторения включений. Предупреждение ошибок повторения включений