Экзаменационная программа дисциплины
«Объектно-ориентированное программирование»
для студентов групп 333, 335А и 335Б
Транспортно-технологического факультета
Сыктывкарского Лесного Института,
2024 – 2025 учебный год, 5 семестр

Макаров П. А., доцент кафедры ИСиТ

Сыктывкар, 2024

Содержание

1 Перечень теоретических вопросов

Общее введение

1. Основные понятия и термины.
2. Стандартизация языков. Инструменты разработчика.
3. Основы архитектуры.
4. Основы операционных систем.
5. Основы компьютерных сетей.

Парадигмы

6. Парадигмы как явление. Парадигмы программирования.
7. Обзор основных парадигм. Совместимость C и C++.
8. Объектно-ориентированное программирование.

Абстрактные типы данных и инкапсуляция

9. Объекты, методы и инкапсуляция.
10. Статический полиморфизм.
11. Конструкторы.
12. Ссылки.

13. Модификатор `const` и константные методы.
14. Динамическая память и копирование объектов.
15. Значения параметров по умолчанию. Заголовок класса и его реализация. Области видимости. Инициализация членов класса в конструкторе.
16. Перегрузка операций функциями вне класса. Дружественные функции и классы.
17. Статические поля и методы.

Обработка исключений

18. Ошибочные ситуации и их обработка. Общая идея механизма обработки исключений.
19. Возбужение и обработка исключений.
20. Обработчики с многоточием. Автоматическая очистка.
21. Преобразования типов исключений. Обработка исключений с помощью специального класса.

Наследование и полиморфизм

22. Иерархия типов объектов. Наследование структур данных и полиморфизм адресов.
23. Защита при наследовании и методы.
24. Конструкторы и деструкторы при наследовании.
25. Виртуальные функции и динамический полиморфизм.
26. Чисто виртуальные методы и абстрактные классы. Виртуальность в конструкторах и деструкторах. Наследование ради конструктора.
27. Виртуальный деструктор. Практическое использование динамического полиморфизма.
28. Приватные и защищённые деструкторы. Наследование как сужение множества.
29. Перегрузка функций и сокрытие имён. Вызов в обход механизма виртуальности.
30. Операции приведения типа.
31. Иерархии исключений.

Шаблоны

32. Шаблоны функций.
33. Шаблоны классов.
34. Специализация шаблонов.

2 Примерные экзаменационные задачи

1. Робот может перемещаться в четырех направлениях («`n`» — север, «`w`» — запад, «`s`» — юг, «`e`» — восток) и принимать пять цифровых команд: `0` — остановка, `1` — продолжать движение, `-1` — движение реверсом, `2` — поворот направо, `-2` — поворот налево. Дан символ `C` — исходное направление робота

и целое число N — посланная ему команда. Вывести направление робота после выполнения каждой полученной команды, вплоть до остановки.

2. Выбран тип шахматной фигуры: K — Король, Q — Ферзь, R — Ладья, N — Конь, B — Слон. Даны координаты двух полей шахматной доски x_1, y_1, x_2, y_2 ($x_{1,2}$ — символы a-h или A-H, $y_{1,2}$ — целые числа, лежащие в диапазоне 1-8).

   Требуется проверить истинность высказывания: «Данная фигура за один ход может перейти с одного поля на другое».

3. Во входном потоке содержится текст на английском языке, заканчивающийся точкой (другие символы «.» в тексте отсутствуют). Требуется написать программу, которая будет определять и выводить на экран английскую букву, встречающуюся в этом тексте чаще всего, и количество таких букв. Строчные и прописные буквы при этом считаются неразличимыми. Если искомых букв несколько, то программа должна выводить на экран первую из них по алфавиту.

4. Во входном потоке содержатся фамилии и имена студентов. Известно, что общее количество студентов не превосходит 100. В первой строке вводится количество зарегистрированных студентов N. Далее следуют N строк, имеющих следующий формат:

   <Фамилия> <Имя>

   Требуется написать программу, которая формирует и печатает уникальный логин для каждого студента по следующему правилу: если фамилия встречается первый раз, то логин — это данная фамилия, если фамилия встречается второй раз, то логин — это фамилия, в конец которой приписывается число 2 и т. д.

5. На автозаправочных станциях (АЗС) продается бензин с маркировкой 92, 95 и 98. В городе $\mathcal{N}$ был проведен мониторинг цены бензина на различных АЗС. Напишите программу, которая будет определять для каждого вида бензина сколько АЗС продают его дешевле всего. Первая строка входного потока содержит количество данных N о стоимости бензина. В каждой из последующих N строк находится информация в следующем формате:

   <Компания> <Улица> <Марка> <Цена>

6. В файле содержатся сведения о телефонах всех сотрудников некоторого учреждения. В первой строке сообщается количество сотрудников N, каждая из следующих N строк имеет следующий формат:

   <Фамилия> <Имя> <Отчество> <Отдел> <Должность> <Телефон>

   Требуется написать программу, которая будет выводить на экран информацию, сколько в среднем сотрудников работает в одном подразделении данного учреждения.

7. Входной поток содержит произвольные алфавитно-цифровые символы. Ввод этих символов заканчивается точкой. Требуется написать программу, которая будет печатать последовательность строчных английских букв (`'a'`, `'b'`, ..., `'z'`) из входной последовательности и частоты их повторения. Печать должна происходить в алфавитном порядке.

8. Дан текстовый файл, содержащий информацию о городах мира (название города, название страны, численность населения, площадь). Выполните следующие задания:

   - Найти самый населенный город в заданной стране;
   - Найти город, который имеет наибольшую плотность населения;
   - Распечатать названия всех стран, которые имеют города с населением более 1 000 000 человек;
   - Найти количество городов, которые расположены в заданной стране.

9. Разработайте класс `Date` (Дата). Определите в нём конструкторы и деструктор, перегрузите операцию добавления к дате заданного количества дней, операцию вычитания и сравнения двух дат. Организуйте необходимые методы ввода/вывода.

10. Создайте класс `Rectangle` (Прямоугольник). Определите в нём конструкторы и деструктор, перегрузите операцию пересечения прямоугольников (операция `*`). Реализуйте методы вычисления площади прямоугольника, а также методы ввода/вывода. Перегрузите операции сравнения прямоугольников (по площади).

11. Разработайте класс `Student` (Студент) со структурными свойствами: фамилия, имя, отчество, номер группы, оценки по трём предметам текущей сессии. Напишите методы ввода/вывода. Перегрузите для класса `Student` операции сравнения (по среднему баллу). Примените этот класс для создания массива объектов класса `Student`. Данные в массив загрузите из файла, содержащего информацию о студентах, отсортируйте этот массив по убыванию среднего балла, результат сортировки запишите в другой файл.

12. Для адресации устройств в компьютерных сетях на сетевом уровне используют IP-адреса. Согласно протоколу IPv4 (Internet Protocol version 4) IP-адрес представляет собой четыре байта, которые обычно пишут в виде десятичных чисел, разделённых точками. Логический 32-битный IP-адрес состоит из двух частей: первая идентифицирует сеть, а вторая — устройство в сети. Для идентификации сети используются первые n ($n < 32$) битов IP-адреса, остальные адресуют устройство. Количество битов, задающих адрес сети определяется сетевой маской. Как и IP-адрес, маска состоит из 32 бит. Маска сравнивается с IP-адресом побитно, слева направо. В маске подсети единицы соответствуют сетевой части, а нули — адресу узла.

Разработатйте класс `NetAddress` (Сетевой Адрес). Определите удобным для себя способом его свойства, конструкторы и деструктор. Перегрузите операции поразрядного логического умножения (операция `&`) и сравнения. Организуйте методы ввода/вывода. Напишите программу, вычисляющую адрес подсети компьютера, по его известным IP-адресу и сетевой маске.

13. Как известно, рациональное число — это несократимая дробь вида n/m, где числитель $n \in \mathbb{Z}$, а знаменатель $m \in \mathbb{N}$. Опишите класс `Rational`, реализующий понятие рационального числа. Снабдите класс четырьмя основными действиями арифметики, соответствующими им операциями присваивания (`+=`, `-=`, `*=`, `/=`), функциями преобразования к числу типа `double` и к целому (здесь необходимы две функции: через отбрасывание дробной части, через округление к ближайшему). Напишите набор тестов, демонстрирующих корректность работы данного класса.

14. Разработайте классы `Matrix` (Прямоугольная матрица) и `QMatrix` (Квадратная матрица), которые должны осуществлять стандартные операции матричного исчисления: сложение, вычитание, матричное умножение, умножение на число, транспонирование. Класс `QMatrix` также должен содержать методы вычисления определителя и получения обратной матрицы.

    Объект класса `Matrix` определяется размерностью матрицы и двумерным массивом её элементов. Поведенческие свойства класса определяются операциями матричного исчисления. Так как квадратная матрица — это частный случай прямоугольной матрицы, то структурные и поведенческие свойства класса `QMatrix` идентичны свойствам класса `Matrix`. В связи с этим, в рамках данной задачи реализуйте класс `QMatrix` публичным наследованием класса `Matrix`, добавив в него методы, специфичные для квадратной матрицы. Для обеспечения доступа к свойствам базового класса из производного, поместите их объявление в секцию `protected`.

    Создание объекта класса `Matrix` требует выделения памяти для хранения его элементов. Поэтому класс `Matrix` обязательно должен содержать конструктор копирования, оператор присваивания и деструктор. Кроме того, дополнительно можно определить конструктор по умолчанию и конструктор с параметрами, определяющими размеры матрицы.

15. Спроектируйте иерархию классов «Вагоны пассажирского поезда» с разделением на общие, сидячие, плацкартные, купейные и СВ. Каждый класс вагона должен содержать информацию о количестве мест разных типов (нижнее, верхнее, нижнее боковое, верхнее боковое), о наличии дополнительных услуг. Реализуйте виртуальные методы, позволяющие рассчитать полный доход от эксплуатации вагона. Создайте класс «Пассажирский поезд», который должен хранить список вагонов. С помощью этой системы классов, напишите программу, определяющую доход от одного рейса поезда.

16. Напишите шаблон функции поиска местоположения элемента в массиве.

Тип элементов массива может быть произвольным.

17. Реализуйте шаблон функции поиска максимального и минимального элементов массива. Тип элементов массива может быть произвольным.

Список литературы

[1] *Вайсфельд М.* Объектно-ориентированное мышление.

[2] *Мейер Б.* Объектно-ориентированное конструирование программных систем.

[3] *Пышкин Е. В.* Основные концепции и механизмы объектно-ориентированного программирования.

[4] *Пол А.* Объектно-ориентированное програмирование на C++.

[5] *Буч Г.* Объектно-ориентированный анализ и проектирование с примерами приложений на C++.

[6] *Андрианова А.А., Исмагилов Л.Н., Мухтарова Т.М.* Объектно-ориентированное програмирование на C++.

[7] *Страуструп Б.* Язык программирования C++.

[8] *Липпман С., Лажойе Ж.* C++ для начинающих.

[9] *Столяров А. В.* Введение в язык Си++.

[10] *Подбельский В. В.* Язык Си++.

[11] http://www.cplusplus.com.

[12] *Пилгрим М.* Погружение в Python 3.

[13] https://www.python.org/doc/.

[14] https://docs.python.org.

[15] https://en.wikibooks.org/wiki/Python_Programming.

[16] https://en.wikibooks.org/wiki/Python_Programming/Classes

[17] https://docs.python.org/3/tutorial/classes.html

[18] https://proglib.io/p/python-oop

[19] https://younglinux.info/oopython/oop