Язык программирования Си++

Иванов А.П., Князева О.С.

Семинар 9. Зачет: методика его проведения, теоретический минимум и типовые задания.

Зачет проводится в практикуме и состоит из двух частей: теоретической и практической.

1. Теоретическая часть

Проверка теоретических знаний производится в виде опроса по любым вопросам, входящим в программу второго семестра и теоретический минимум. Вопрос, как правило, формулируется в виде просьбы написать на листе бумаги короткий (2-5 строк) код, иллюстрирующий то или иное понятие языка программирования. Обычно задается от двух до пяти вопросов. Студенты, не сдавшие теоретическую часть, автоматически не допускаются к выполнению практической части зачета.

Ниже приводится список вопросов теоретического минимума знаний, которые студент должен иметь для получения зачета:

1. Что такое: инкапсуляция, наследование, полиморфизм? Пояснить механизм реализации каждого из принципов объектно-ориентированного программирования в синтаксисе языка программирования Си++.
2. Классы. Конструкторы, деструктор. (Определить класс, определить конструктор и деструктор в нем). Может ли в классе быть несколько деструкторов? Несколько конструкторов?
3. Перегрузка операторов и функций. Особенности перегрузки и возврата значений в операторах «=» и «+». Перегрузка оператора «+=».
4. Перегрузка операторов ввода и вывода, работающих с потоками Си++.
5. Перегрузка оператора приведения типа.
6. Оператор «`::`». Определение тела метода вне класса.
7. Полиморфизм, виртуальные функции. Проиллюстрировать разницу в работе обычного и виртуального методов.
8. Виртуальный деструктор. Зачем применяется?
9. Динамическая память. Операторы `new` и `delete`. Выделить память под 100 вещественных чисел, потом ее освободить.
10. Дружественные классы и функции (`friend`). Зачем применяются?
11. Значения аргументов функций по умолчанию. Как задать? Всегда ли можно ли задать по умолчанию первый аргумент функции?
12. Одиночное наследование. Проиллюстрировать работу унаследованного, переопределенного и нового методов в производном классе.
13. Инкапсуляция. Права доступа к членам класса: `private`, `protected`, `public`. Проиллюстрировать разницу.
14. Права доступа при наследовании класса: `private`, `protected`, `public`. Проиллюстрировать разницу.

15. Множественное наследование.
16. Чисто виртуальные функции. Что такое? Зачем применяются?
17. Абстрактные классы.
18. Передача и возврат параметров в функции по значению, по ссылке, по указателю.
19. Потоки ввода–вывода Си++ `cin` и `cout` и их использование. Манипуляторы потоков `endl`, `flush`. Смена формата вывода целых и вещественных чисел.
20. Потоки ввода–вывода Си++, определяемые в заголовочных файлах: `<iostream>`, `<fstream>`, `<sstream>`. Назначение, особенности использования.
21. Файловый ввод–вывод в Си++. Открытие потока, связанного с файлом на чтение, запись. Закрытие потока.
22. Ссылки. Чем отличаются от указателей? Нужно ли инициализировать ссылки при их объявлении? Работа оператора присваивания со ссылкой. Можно ли вернуть ссылку из функции на переменную, объявленную в этой функции?
23. Статические методы и поля класса.
24. Указатели. Операторы взятия адреса и взятия значения по адресу.
25. Пространства имен (`namespace`, `using`). Пространство имен `std`.
26. Обобщенное программирование. Шаблоны функций.
27. Обобщенное программирование. Шаблоны классов.
28. Строковые классы Си++ (`string`) и их использование. Инициализация значения, определение длины строки, поиск символа, поиск подстроки. Присваивание строк.
29. Библиотека стандартных шаблонов (STL): контейнер `vector`. Инициализация, определение размера, считывание значения элемента, изменение значения элемента.
30. Библиотека стандартных шаблонов (STL): контейнер `list`. Инициализация, определение размера, считывание значения элемента, изменение значения элемента.
31. Библиотека стандартных шаблонов (STL): контейнер `set`. Инициализация, определение размера, считывание значения элемента, изменение значения элемента.
32. Библиотека стандартных шаблонов (STL): контейнер `map`. Инициализация, определение размера, считывание значения элемента, изменение значения элемента.
33. Библиотека стандартных шаблонов (STL): поиск элемента в контейнере и сортировка элементов контейнера.
34. `Explicit`-конструкторы. В каких ситуациях применяются?
35. Обработка исключений Си++. Ключевые слова `try`, `throw`, `catch`, и их использование.

2. Практические задания

Практические навыки программирования проверяются на одной типовой задаче, подобной тем, что выполнялись в течение семестра. Студент должен ее выполнить в течение одной пары (два академических часа) от начала до конца.

Рекомендуется основную массу заданий выдавать по образцу семинаров 1, 3, 6, 7:

- Создание класса, перегрузка арифметических операторов, ввод-вывод.
- Взаимодействие классов: матрицы и вектора. Динамическая память.
- Наследование, виртуальные функции.
- Создание шаблонного класса.
- Контейнеры библиотеки стандартных шаблонов.
- Работа с файлами.

3. Примеры практических заданий

1. Создайте класс систем из трех линейных уравнений с тремя неизвестными вида `Ax=B` (где `A` – квадратная матрица `nxn`, `x` – столбец из `x1, x2 ... xn`, `B` – это столбец `b1... bn`) . Реализуйте в нем метод решения данной системы методом Гаусса. Метод Гаусса состоит в приведении с помощью элементарных операций (умножение строки на число, сложение, вычитание перестановка строк) системы уравнений к системе с треугольной матрицей, т.е. к системе вида

$$\begin{cases} x_1 + c_{12}x_2 + \ldots + c_{1n}x_n = d_1, \\ \quad x_2 + \ldots + c_{2n}x_n = d_2, \\ \ldots \\ \quad x_n = d_n, \end{cases}$$

   которая уже решается тривиально.

2. Создать функцию обрабатывающую текстовый файл. В файле записаны выражения вида:

```
sin’(2*x) =
tg’(1/x) =
```

   Функция должна распознавать записанные тригонометрические функции, брать производную и создать новый файл, вида:

```
sin’(2*x) = 2*cos(2*x)
tg(1/x) = -1/x^2*1/cos^2(1/x)
```

   В файле могут быть записаны только тригонометрические функции `sin`, `cos`, `tg`, `ctg` от выражений вида (`a/x^b` или `a*x^b`). Функция принимает имена входного и выходного файлов. Работа с файлом должна осуществляться с помощью функций библиотеки `<fstream>`.

3. Создайте класс, работающий с конфигурационными файлами вида:

```
Temperature = 273.0
Pressure = 1e+5
N = 100
kB = 1.38e-23
Save result = true
```

   Реализуйте в нем метод открытия файла. Создайте дружественную шаблонную функцию, которая в качестве аргумента принимает строку – название параметра в конфигурационном файле, возвращает его значение (которое может быть, целым, вещественным или булевским типом). Работа с файлом должна осуществляться с помощью функций библиотеки `<fstream>`.

4. Создать функцию обрабатывающую текстовый файл. В файле записаны выражения вида:
```
sin(3.5) =
ctg(1.2) =
```
Функция должна вычислить результат каждого выражения и создать новый файл вида:
```
sin(3.5) = -0.3508
ctg(1.2) = 0.3888
```
В файле могут быть записаны только тригонометрические функции `sin`, `cos`, `tg`, `ctg`. Функция принимает имена входного и выходного файлов. Работа с файлом должна осуществляться с помощью функций библиотеки `<fstream>`.

5. Создать функцию обрабатывающую текстовый файл. В файле записана таблица (произвольной размерности) пар значений (x_i и y_i) некоторой физической величины, при этом значения x_i – упорядочены по возрастанию. Пользователь вводит некоторое значение x, функция должна вернуть соответствующий этому значению y, который должен быть получен методом линейной интерполяции табличных значений.
Функция принимает имена входного файла и величину x. Работа с файлом должна осуществляться с помощью функций библиотеки `<fstream>`.

6. Создать класс матриц произвольного (но не изменяемого для непустой матрицы) размера, реализовать в нем методы транспонирования матрицы, считывания матрицы из файла, записи матрицы в файл.

7. Создать калькулятор для класса матриц произвольного размера, записанных в файле. реализовать для матриц методы сложения, вычитания, умножения, присваивания, пользователь указывает два входных файла, знак операции и выходной файл.

8. Создать классы: трехмерный вектор и матрица 3x3. Определить операции сложения, вычитания, умножения на число, скалярного произведения и произведения матрицы на вектор и вектора на матрицу.

9. Создать классы: 6-мерный вектор и производный от него класс: симметричная матрица 3x3. Определить для них операции присваивания, сложения, вычитания, умножения на число, умножения, при этом методы матрицы должны максимально использовать соответствующие методы базового класса.

10. Создать класс матриц произвольного размера. Реализовать в нем методы удаления заданной строки или столбца.

11. Создайте класс n-мерных векторов. Определите в нем конструктор, деструктор, конструктор копирования. Перегрузите операции `[]` (получении i – координаты вектора), = (присваивание). Перегрузите операции ввода и вывода. Сделайте метод дополнения вектора новым элементом, записываемым в конец вектора.

12. Создайте класс рациональных чисел. Определите в нем конструктор, деструктор, конструктор копирования Перегрузите операции + , - , * , / , = , операции ввода и вывода. Перегрузите операцию приведения типов к вещественным числам и комплексным числам.

13. Создайте класс вещественных чисел, которые представлены в виде $a \cdot 10^b$, где a – это мантисса, b – это экспонента, т.е., например, число 668.25 должно храниться в виде двух чисел a=0.66825, b=3. Определите в классе конструктор, деструктор, конструктор копирования Перегрузите операции + , - , * , / , = , операции ввода и вывода. Перегрузите операцию приведения типов к вещественным числам.

14. Создать свой класс строк, реализовать строку как динамический символьный массив. Определить в классе конструктор, деструктор, конструктор копирования. Перегрузить

в нем операции + , = , операции ввода и вывода. Создать метод класса: поиск подстроки.

15. Создать свой класс строк `str` из STL `vector<char>`. Перегрузить в нем операции +, =, сравнения, ввода и вывода. Реализовать методы класса – вставка подстроки на заданную позицию, удаление подстроки, преобразование строки `str` в строку Си.
16. Создайте абстрактный класс – маятники. Определите в нем чисто виртуальную функцию – период колебаний. Создайте на его базе 2 производных класса – математический маятник и пружинный маятник. Создайте массив, которые содержит указатели на объекты этих классов. Напишите функцию, определяющую маятник в массиве с максимальным периодом колебаний.
17. Создайте базовый класс точек в трехмерном пространстве. Переопределите для них операции ввода, вывода. Создайте два производных класса трехмерных векторов в декартовых и полярных координатах. Определите для них операции сложения, скалярного и векторного произведения, метод определения длины вектора.
18. Создайте абстрактный класс геометрический объект (с чисто виртуальной функцией площадь объекта) и несколько производных классов (квадрат, круг, треугольник). Создайте класс список геометрических объектов и определите в этом классе метод – общая площадь объектов.
19. Создайте шаблонный класс матриц на базе STL класса `vector`. Определите в нем конструкторы и деструктор. И два метода класса – заполняющих заданный `std::vector` указанным столбцом или строкой текущей матрицы.
20. На базе `std::vector` создать свой шаблонный класс: контейнер ключей (целых чисел) `myset` (все ключи уникальны и расположены в контейнере по возрастанию). Определить в нем функцию поиска по значению и вставки (с учетом того, что ключи должны быть уникальны и вставка должна происходить таким образом, чтобы контейнер оставался отсортированным).
21. Определите два класса комплексных и рациональных чисел. Переопределите для них операции сравнения и равенства. Создайте шаблонные функции: максимум из двух переменных `max(a,b)` и функцию `swap(a,b)`, которая присваивает переменной *a* значение *b*, а переменной *b* значение *a* и примените их для объектов ваших классов.
22. Создайте класс книг и производный к нему класс библиотечных книг (с указанием количества обращений). Создайте класс каталог библиотечных книг – как динамический массив и реализуйте в нем методы поиска наиболее и наименее популярной книги.
23. Создайте класс: вектор произвольной размерности с операцией вставки нового элемента в любой позиции. Создайте производный класс: «сортированный вектор» с методом добавления нового элемента, который бы использовал метод вставки базового класса.
24. На базе двух `std::vector` создать свой шаблонный класс контейнер mymap (один массив – массив ключей, второй массив – массив значений). Определить в нем функцию поиска по значению и вставки (с учетом того, что ключи должны быть уникальны).
25. Создайте шаблонный класс матриц. Определите в нем конструкторы и деструктор. Определите дружественные функции сложение и умножение матриц. Проиллюстрировать работу для целых и вещественных матриц.