Список заданий по курсу «ООП на C++»

А. Г. Фенстер

31 августа 2010 г.

Для получения зачёта по практике необходимо сдать все либо некоторые из перечисленных ниже заданий. Оценка за практику будет выведена следующим образом:

- за несданные задания студент получит 0 баллов;
- за каждое из первых трёх заданий — 3, 4 или 5 баллов;
- за четвёртое задание — 1 или 2 балла, при этом четвёртое задание необходимо сдать всем;
- итоговая оценка будет определена по следующей таблице:

кол-во баллов	0 – 6	7 – 11	12 – 14	15 – 17
оценка	неудовл.	удовлетв.	хорошо	отлично

Существуют также некоторые правила, которые необходимо выполнять, а именно:

1. Нельзя сдавать чужой код. Санкции за нарушение могут быть любыми, вплоть до замены задачи или 0 баллов за эту задачу. Плагиат отслеживается автоматической системой!
2. Студент может досдавать части задачи для повышения оценки (например, сначала сдать все задания на «тройки», а потом дописывать код для получения более высоких оценок). При этом, если задача была сдана на какую-либо оценку, штрафные санкции за опоздание к этой задаче уже не будут применяться.
3. Задания необходимо сдавать вовремя! **За опоздание снимаются баллы: по 1 баллу за каждые две недели опоздания** (до достижения минимально возможной оценки за задание).
   Срок сдачи каждой задачи — конец очередного месяца.

Задание 1. Строки

Реализуйте класс для хранения строк и безопасной работы с ними (аналог `std::string`). Каждый объект класса должен хранить значение типа `char *` и самостоятельно отслеживать, какое количество памяти необходимо для хранения строки.

Пример такого класса:

```
class String
{
     private:
          char *data;
     ...
};
```

Возможно, вам захочется также завести приватное поле для хранения длины строки или указателя на последний элемент.

Необходимо реализовать:

- конструктор по умолчанию (создаёт пустую строку);
- конструктор с параметром `char *`;
- конструктор копирования;
- деструктор (освобождает выделенную память);
- оператор присваивания;
- оператор приведения к `const char *`;
- тестовую программу (функцию `main`), проверяющую всё перечисленное;
  (теория по классам, конструкторам, деструкторам и т. п.)

на «тройку»

- конкатенацию двух строк: операторы `+` и `+=`;
- проверку этих операторов;
  (теория по операторам)

на «четвёрку»

- обращение по индексу — два варианта оператора `[]`: константный для чтения, возвращающий `char &` для изменения;
- вставить в оба оператора отладочный вывод и продемонстрировать, в каких случаях вызывается каждый из них.

на «пятёрку»

Задание 2. Список

Реализуйте шаблон класса для хранения динамического списка элементов типа `T` (аналог `std::list`).

Пример:

```
template <typename T> class List
{
     private:
          struct element
          {
               T data;
               element *next;
          };
          element *head;
     ...
};
```

Необходимо реализовать:

- операции вставки элемента в начало списка;
- операцию удаления первого элемента;
- деструктор, освобождающий всю выделенную память;
- тестовую программу, проверяющую всё вышеперечисленное;
  (теория по шаблонам)

на «тройку»

- операции вставки элемента в конец списка;
- операцию удаления последнего элемента;
- запрет использования конструктора копирования и оператора присваивания;
- тестовую программу для всех этих операций;

на «четвёрку»

- класс-итератор, перебирающий элементы списка при вызове оператора `++`, для которого определены операторы `*` и `->`;
- тестовую программу, использующую этот класс.
  (теория по операторам `*` и `->`)

на «пятёрку»

Задание 3. Символьное дифференцирование

Реализуйте абстрактный класс `Expression` с чисто виртуальными методами `Expression *diff();` и `void print();` и отнаследуйте от него классы `Number`, `Variable`, `Add` и `Sub` — число, переменная, сумма двух выражений и разность двух выражений, для которых напишите реализацию этих двух методов.

Пример:

```
Expression *e = new Add(new Number(1), new Variable('x'));
std::cout << "\n";
Expression *de = e->diff();
de->print();
std::cout << "\n";
delete e;
delete de;
```

Результатом выполнения этого кода должно быть
`0+1`

Естественно, не запрещается сделать упрощение выражений и научить `diff()` дифференцировать по заданной переменной, а не только по `x`.

Необходимо реализовать:

- классы `Expression`, `Number`, `Variable`, `Add`, `Sub`.
  (теория по наследованию)

на «тройку»

- классы `Mul` (произведение), `Div` (частное), при желании функции типа `Sin`, `Cos` и т.п.

на «четвёрку»

- чтение выражения из строки или `std::cin` с автоматическим созданием необходимых объектов. Пример:

  ```
  Expression *e = read_expression("((x+1)*x)");
  ```

  Допускается требовать наличие скобок вокруг операндов любой бинарной операции (это позволит написать разбор выражения при помощи очень простого рекурсивного спуска).

на «пятёрку»

Задание 4. График функции

Целью задания является освоение какой-либо графической библиотеки и написание программы, отрисовывающей график некоторой функции $f(x)$. На занятиях будет показано, как «рисовать» в окне приложения Windows, используя методы класса `CPaintDC`.

Вы не ограничены в выборе графической библиотеки для рисования, но рисование на `CPaintDC` является, пожалуй, одним из самых простых способов.

Общие условия:

- функция для рисования задаётся прямо в тексте программы;
- не требуется никакого красивого оформления, достаточно нарисовать оси координат и график функции;
- график функции рисуется отрезками (`dc.MoveTo(x, y)`, `dc.LineTo(x, y)`), а не точками;
- функция предполагается непрерывной и ограниченной на интересующем нас интервале.

Задание необходимо сделать и сдать всем!

- рисование графика заданной в коде программы функции.
- позиционирование графика по центру окна. Узнать размеры окна можно, например, так:

```
RECT rect;
dc.GetWindow()->GetWindowRect(&rect);
```

на 1 балл

- используя классы из задания 3, нарисовать график заданной при помощи классов `Add`, `Mul`, ... функции и график её производной. Для этого к классам задания 3 необходимо добавить виртуальный метод `double evaluate(double x)`, вычисляющий значение функции в точке `x`.

на 2 балла