Домашняя работа №1

Задание 1. Реализуйте класс `Complex` для работы с комплексными числами. В качестве базового вещественного типа используйте **`double`**. В классе определите все необходимые конструкторы и операции, в том числе операцию чтения из потока, которая считывает два подряд идущих (разделители — пробельные символы) числа типа **`double`** в объект класса `Complex`.

С использованием класса `Complex` напишите программу, которая вычисляет, принадлежит ли заданная точка *Множеству Мандельброта*. Множество Мандельброта — это множество M комплексных чисел, состоящее из точек m, для которых последовательность

$$\begin{aligned} z_0 &= 0 \\ z_{i+1} &= z_i^2 + m \end{aligned}$$

сходится к 0. Необходимым условием принадлежности точки множеству является условие $|z_i| \leqslant 2$ для любой итерации i. На практике множество Мандельброта вычисляют приближённо следующим образом: положим K — максимальное количество итераций. Если за K итераций (то есть до z_K включительно) необходимое условие не было нарушено, точка считается принадлежащей множеству.

Ваша программа должна считывать со стандартного потока ввода сначала целое число K ($1 \leqslant K \leqslant 10000$), затем последовательность комплексных чисел до признака конца файла. Для каждого комплексного числа напечатайте на стандартный поток вывода 0, если число принадлежит множеству, и номер итерации k, для которого первый раз было нарушено необходимое условие принадлежности точки множеству, если число не принадлежит множеству.

Задание 2. Напишите класс `Integer`, который реализует арифметику с целыми числами (типа **`int`**) с контролем переполнения при выполнении операций. При возникновении переполнения должно выбрасываться исключение `overflow_error`. При попытке деления на 0 или взятия остатка от деления на 0 должно выбрасываться исключение `divide_by_zero_error`. В классе реализуйте все необходимые конструкторы, арифметические операции, операторы вывода в поток и ввода из потока.

Рассмотрим следующую программу, которая вычисляет числа Фибоначчи.

```
#include <stdio.h>

int fib(int n)
{
  if (n <= 1) return n;
  return fib(n - 1) + fib(n - 2);
}

int main(void)
{
  printf("%d\n", fib(42));
  return 0;
}
```

Вследствие неэффективного алгоритма функция `fib` работает экспоненциальное время. Например, данная программа на процессоре Pentium-4 1.8G работает примерно 11 секунд.

- Перепишите эту программу на Си++ так, чтобы она использовала класс `Integer`. Как изменилось время её работы?

- Перепишите эту программу на Си++ таким образом, чтобы результат работы функции возвращался с помощью исключений. Как изменилось время её работы?
- Совместите два предыдущих пункта. Каково время работы получившейся программы?
- Перепишите функцию `fib` эффективно, то есть реализуйте вычисление чисел Фибоначчи в цикле. Как изменилось время работы программы.
- Замените функцию `main` Вашей программы таким образом, чтобы она считывала данные со стандартного потока ввода и выводила результат на стандартный поток вывода. Данные считываются до признака конца ввода. Каждый новый результат печатается на отдельной строке. Для реализации ввода-вывода используйте потоки Си++.

В качестве решения задания 1 будет приниматься программа, реализующая последний пункт. В комментарии в начале Вашей программы приведите время работы Вашей программы в исходном варианте, с классом `Integer`, с исключениями, и с классом `Integer` и с исключениями и эффективного варианта. Укажите какой компилятор Си++ использовался и на какой машине запускалась программа.