Programowanie 2

Zadanie 2

Piotr Błaszyński

12 marca 2019

Na podstawie **kopii** kodu z zadania z pierwszych zajęć (UWAGA! korzystamy z wersji jakie mamy, nie staramy się dopisywać rozwinięcia dla np. większych liczb) przygotować wersję programu posługującą się klasą i obiektami tej klasy. Kod z poprzednich zajęć proszę przechowywać oddzielnie. Nazwy zmiennych, klas i obiektów mają być po angielsku. Klasę opisujemy przy pomocy słowa kluczowego class, nazwy klasy i definicji klasy zaczynającej się i kończącej nawiasem klamrowym i średnikiem. Definicja składa sie listy metod i atrybutów klasy oraz ewentualnego opisu dostepności pól. Przykładowo poniższa definicja:

```
class Complex
{
public://sekcja publiczna to jest czesc kodu, ktora jest dostepna dla
    innych klas i metod
        Complex(double, double);
        ~Complex();
        string ToString();
private://sekcja prywatna to jest czesc kodu, ktora nie jest dostepna
    dla innych klas
        double real, imaginary;
};
```

opisuje klasę o nazwie Complex, z publicznym konstruktorem przyjmującym 2 parametry typu double. Ponadto klasa ta posiada metodę toString oraz publiczny destruktor. Do wykonywania swoich zadań klasa używa 2 pól prywatnych.

Ponadto, aby zdefiniować treść metody należy posłużyć się następującym kodem:

```
Complex::Complex(double _real, double _imaginary)
{
        real = _real;
        imaginary = _imaginary;
}
```

lub:

```
string Complex::ToString()
{
        stringstream s;
        s << real << " + " << imaginary << "j";
        return s.str();
}
```

Obiekty tak opisanej klasy tworzymy w następujący sposób:

```
Complex firstObject(1, 5);
Complex secondComplexObject(1, 5);
Complex thirdComplex(1, 5);
```

Powyższych obiektów używa się np. w taki sposób:

```
std::cout << firstObject.ToString() << std::endl;
std::cout << secondComplexObject.ToString() << std::endl;
std::cout << thirdComplex.ToString() << std::endl;
```

Minimalna implementacja ma umożliwiać utworzenie obiektu klasy służącej do wykonywania przeliczenia na wersję cyfrową. Następnie na rzecz tego obiektu należy wywołać metodę, która zwróci nam wersję zapisaną za pomocą cyfr. Program ma zawierać utworzenie przynajmniej kilku obiektów z różnymi wartościami liczbowymi zapisanymi przy pomocy cyfr i liter i pokazanie na ekranie ich wersji zapisanej przy pomocy cyfr. **Sama klasa nie może** pobierać nic z klawiatury ani wypisywać na ekran.

Dla aspirujących na wyższe oceny: Prosze przygotować klasę testującą Assert, której obiekt będzie można wykorzystać w taki mniej więcej sposób:

```
BigNumber trillionBig("1T");
BigNumber speedBig("3M");
std::cout << assert.AreEqual("1 000 000 000 000", trillionBig.
    getValue()) << std::endl;
std::cout << assert.AreEqual("300 000", speedBig.getValue()) << std::
    endl;
```

i otrzymamy wyniki 1 i 0 (metoda AreEqual zwraca 1 jeżeli argumenty są równe, 0 jeżeli nie są). Umieścić w funkcji main kilka wywołań dla różnych wartości testowych.