Klasy

Do tej pory w programach używaliśmy wyłącznie zmiennych różnych typów. Na ich bazie tworzyliśmy również funkcje, które wykonywały różne zadania. Obecnie zajmiemy się niejako tworzeniem nowego typu danych, a mianowicie klasami.

Klasa (ang. class) – nowy typ danych, której egzemplarzami są obiekty.

Poniżej przedstawimy przykład klasy o nazwie `Prostokat`. Będzie ona zawierać wszystkie podstawowe elementy, które są charakterystyczne dla dowolnej klasy. Następnie omówimy znaczenie tych elementów.

Zadanie 1.
Stworzyć plik `klasa1.cpp` i wpisać w nim następujący kod:

```
#include <iostream>
using namespace std;

class Prostokat
{
   private:
   int szerokosc,wysokosc;

   public:
   Prostokat(int x, int y);
   int Pole();
   int Obwod();
   void wypisz();
   void ustawSzerokosc(int sz);
   int wezSzerokosc ();
      };
   Prostokat::Prostokat(int x, int y)
     {
        szerokosc=x;
        wysokosc=y;
     }
   int Prostokat::Pole()
      {
         return (szerokosc*wysokosc);
      }

   int Prostokat::Obwod()
      {
          return (2*szerokosc+2*wysokosc);
      }

   void Prostokat::wypisz()
      {
       cout<<"Szerokosc wynosi "<<szerokosc<<" ,a wysokosc "<<wysokosc<<endl;
       cout<<"Pole wynosi "<<Pole()<<" a obwod "<<Obwod()<<endl;
      }

   void Prostokat::ustawSzerokosc(int sz)
      {
```

```
       szerokosc=sz;
      }

   int Prostokat::wezSzerokosc()
   {
    return szerokosc;
   }

int main()
{
    int a,b;

     cout<<"Podaj szerokosc prostokata a=";
     cin>>a;
     cout<<"Podaj wysokosc prostokata b=";
     cin>>b;
  Prostokat prostokat_1(a,b);
    //Prostokat prostokat_2(2,3);
  prostokat_1.wypisz();
    // prostokat_2.wypisz();
  prostokat_1.ustawSzerokosc(20);
  cout<<prostokat_1.wezSzerokosc()<<endl;
  prostokat_1.wypisz();
   system("pause");
   return 0;
}
```

następnie skompilować plik oraz uruchomić program.

Ogólnie szablon klasy wygląda następująco:

```
class NazwaKlasy
  {
      atrybuty, metody;
      };
```

Zaczynamy od słowa kluczowego class, następnie piszemy nazwę klasy (w naszym przypadku nazwa klasy to `Prostokat`) oraz definicję klasy składającą się ze zmiennych i funkcji, które w przypadku klas nazywają się odpowiednio: atrybuty i metody. Umieszczamy je w obrębie nawiasów klamrowych. **Na końcu definicji umieszczamy średnik.** Definicję klasy, podobnie jak funkcji, umieszczamy przed funkcją `main()`. W naszym przypadku atrybutami klasy `Prostokat` są: `szerokosc` oraz `wysokosc` – obie zmienne typu całkowitego. Są one umieszczone po słowie kluczowym `private:`, które oznacza, że dostęp do tych danych jest prywatny. Powoduje to, że użytkownik nie może zmienić tych danych w programie w tradycyjny sposób, czyli poprzez przypisanie innej wartości. Metody w naszej klasie to: `Prostokat(int x, int y)`, `Pole()`, `Obwod()`, `wypisz()`, `ustawSzerokosc(int sz)`, `wezSzerokosc()`. Dostęp do nich jest publiczny (umieszczamy je po słowie kluczowym `public:`). Na tym kończy się definicja klasy. Następnie definiujemy poszczególne metody. Zauważmy, że w definicji tych funkcji nazwę każdej z nich poprzedzamy nazwą klasy do której należą oraz operatorem zasięgu `::`. Jest to konieczne, gdyż możemy definiować kilka klas, które zawierają metody o tych samych nazwach i wówczas taki właśnie zapis w definicji pozwoli stwierdzić do której klasy dana funkcja należy. Omawianie poszczególnych metod zaczniemy od pierwszej, która różni się od pozostałych. Nosi ona nazwę **konstruktor**. Cechą charakterystyczną konstruktora jest to, że jego nazwa jest taka sama jak nazwa klasy, a więc `Prostokat`. Ponadto nigdy nie zwraca żadnej wartości (dlatego jest typu `void`). Konstruktor służy do „powoływania obiektu do życia”. Wytłumaczymy to na przykładzie naszej klasy. Zacznijmy od definicji. Konstruktor `Prostokat` pobiera dwa argumenty – liczby całkowite `x,y`, które są przypisywane odpowiednio do zmiennych `szerokosc, wysokosc`. Można więc powiedzieć, że konstruuje on obiekt, czyli prostokąt o bokach długości `x,y`. Następnie jest on wywoływany w programie (wewnątrz funkcji `main()`) poleceniem

```
Prostokat prostokat_1(a,b);
```

Tworzony jest więc obiekt typu naszej klasy (egzemplarz klasy), czyli typu `Prostokat` o nazwie `prostokat_1`, którego szerokość wynosi `a`, zaś wysokość wynosi `b`.
Metody `Pole()` oraz `Obwod()` służą do obliczania pola i obwodu prostokąta. Nie muszą one pobierać żadnych argumentów, gdyż są składowymi klasy, która zawiera również atrybuty `szerokosc` i `wysokosc`.
Metoda `wypisz()` służy do wypisania informacji ile wynosi pole, a ile obwód prostokąta.
Metoda `ustawSzerokosc(int sz)` pozwala ustawić nową wartość szerokości prostokąta równą `sz`. Metoda `wezSzerokosc()` z kolei zwraca zmienioną przy pomocy poprzedniej funkcji wartość zmiennej `sz`.
Sposób wywołania tych funkcji w programie (ogólnie wszystkich funkcji) jest następujący:

```
Nazwa_obiektu.nazwa_metody();
```

U nas, chcąc przykładowo wypisać informację o polu i obwodzie prostokąta, użyjemy składni:

```
prostokat_1.wypisz();
```