Programowanie obiektowe C++

Obiekty, tablice obiektów

Obiekty, tablice obiektów

1. Przykład

Poniższy przykład ilustruje pracę z obiektami opisującymi studentów.

```
#include <iostream>
#include <string>

using namespace std;

class TStudent
{
  public:
    TStudent(string surname, string name, int studentId, int year);

    string getFullName() const;
    int getStudentId() const;

  private:
    string surname;
    string name;
    int studentId;
    int year;
    int courseCount;
    string courses[16];
};

TStudent::TStudent(string surname, string name, int studentId, int year)
: surname(surname), name(name), studentId(studentId), year(year), courseCount(0)
{
}

string TStudent::getFullName() const
{
  string str = name + " " + surname;

  return str;
}

int TStudent::getStudentId() const
{
  return studentId;
}

int main()
{
  TStudent stud("Kowalski", "Albert", 1, 1);
```

```
  cout << stud.getFullName() << endl;
  cout << stud.getStudentId() << endl;

  return 0;
}
```

Wprowadź program i przetestuj jego działanie dla przykładowych danych.

2. Zadanie

Uzupełnij powyższy kod o następujące metody:

- konstruktor pozwalający wpisywać nazwy przedmiotów do tablicy **`courses[]`**,
- **`getAll()`** – zwraca łańcuch znaków ze wszystkimi informacjami o studencie (włącznie z nazwami przedmiotów); bez argumentów,
- **`setStudentId()`** – pozwala na zmianę numeru indeksu studenta; argumentem jest nowy numer indeksu, zwraca **`void`**,
- **`getYear()`** – pozwala na sprawdzenie roku studiów studenta; bez argumentów, zwraca numer roku,
- **`nextYear()`** – inkrementuje rok studiów o jeden; zwraca **`void`**,
- **`addCourse()`** – dodaje nazwę przedmiotu do tablicy, zwraca **`void`**,
- **`removeCourse()`** – usuwa przedmiot i na jego miejsce przenosi ostatni przedmiot w tablicy; zwraca **`void`**.

Do napisania powyższych funkcji może być przydatna przeciążona funkcja **`std::to_string()`**, zamieniająca wartości różnych typów na łańcuch znaków oraz zdefiniowany w przestrzeni nazw **`std`** przeciążony operator relacji **`==`**, który można stosować z łańcuchami znaków.

Zadania dodatkowe

1. Program z deklaracją obiektu **`TSquare`**:

```
class TSquare
{
  public:
    float perimeter();
    float area();

  private:
    float side;
};
```

Uzupełnij o:

- odpowiednie zadeklarowane metody,
- metody pozwalające na odczyt i zmianę boku kwadratu,
- domyślny konstruktor klasy (bez parametrów lub z domyślnymi parametrami),
- zdefiniuj tablicę obiektów (np.: **`TSquare squares[5]`**) i zmień niektóre wielkości boków kwadratów,
- wykonaj obliczenia pola i obwodu po wszystkich elementach tablicy w pętli **`for()`**.