Podsumowanie mini projektów z tematu konstruktor, destruktor, przeładowanie funkcji

1. Jeżeli tworzycie Państwo obiekty dynamicznie (za pomocą operatora **new**), to pamiętajcie Państwo o zwalnianiu przydzielonej pamięci (za pomocą operatora **delete**).

2. **Konstruktor domniemany** to taki, który MOŻEMY wywołać BEZ ŻADNYCH ARGUMENTÓW.
3. Konstruktor domniemamy może mieć dowolną ilość domniemanych parametrów.

4. Konstruktor domniemamy AUTOMATYCZNIE generowany jest TYLKO jeżeli w klasie NIE ZADEKLARUJEMY żadnego KONSTRUKTORA.

5. Jeżeli w klasie zadeklarujemy konstruktor/różne konstruktory (w tym NIE zadeklarujemy domniemanego konstruktora) to możemy „wymusić" na kompilatorze stworzenie domniemanego konstruktora (funkcjonalność wprowadzona przez standard C++11):

```
class Tklasa
{
  Tklasa()=default;
};
```

//wówczas kompilator utworzy konstruktor domniemamy, nawet jeśli w kodzie napisaliśmy inne konstruktory

6. Jeżeli chcecie Państwo wymusić na kompilatorze NIE tworzenie konstruktora domniemanego, nawet jeżeli w klasie nie stworzyliście żadnego konstruktora, wówczas należy to zapisać w ten sposób:

```
class Tklasa
{
  Tklasa()=delete;
};
```

//wówczas kompilator nie utworzy konstruktora domniemanego, nawet jeśli w kodzie nie napisaliśmy żadnego konstruktora

7. Przeładowanie nazwy funkcji oznacza utworzenie wielu funkcji o tej samej nazwie, różniących się typem i ilością argumentów.

8. Konstruktor może być przeładowany. Co więcej jest to najczęściej przeładowywana funkcja. Często w klasie potrzebne nam są różne konstruktory, które będą uruchomione w zależności od przypadku.

W odniesieniu do zadań:

Polecenie 1: Do dynamicznego tworzenia obiektu w języku C++ wykorzystujemy operator *new*.

Proszę pamiętać o zwalnianiu dynamicznie przydzielonej pamięci za pomocą operatora *delete*.

Polecenie 3: W tym poleceniu chodziło mi o to, żebyście Państwo NIE tworzyli sami konstruktora domyślnego, tylko wymusili na kompilatorze jego utworzenie za pomocą polecenia:

```
class TStudent
{
        TStudent() = default; //kompilator ma „przywrócić” niejako „ustawienia fabryczne”
};
```

Polecenie 5: Mieliście Państwo stworzyć konstruktor kopiujący (w parametrze otrzymuje oryginalny obiekt ale nie dokonuje żadnych modyfikacji tego oryginalnego obiektu). Natomiast w funkcji main mieliście Państwo stworzyć dwa obiekty, przy czym jeden dynamicznie, a zatem operatorem new. Zaś do drugiego obiektu należało za pomocą wskaźnika skopiować dane obiektu 1:

```
class TOsoba
{

private:
        string imie;
        int wiek;
public:
        TOsoba(const TOsoba &osoba)
        {
                imie = osoba.imie;
                wiek = osoba.wiek;
                cout << "konstruktor kopiujacy" << endl;
        }

        TOsoba(string _imie = "", int _wiek = 0)
        {
                imie = _imie;
                wiek = _wiek;
        }

        void Napis();

};

void TOsoba::Napis()
{
        cout << "Imie" << imie << endl;
        cout << "wiek" << wiek << endl;


}

int main()
```

```
{
        TOsoba *osoba1 = new TOsoba("Karol", 22);
        TOsoba osoba2(*osoba1);

        osoba1->Napis();
        osoba2.Napis();

        delete osoba1;

        system("pause");
}
```