Konstruktory, przeładowanie funkcji, funkcja inline

1. **Przeładowanie (ang. overloading) nazwy funkcji** – w danym zakresie ważności jest więcej niż jedna funkcja o tej samej nazwie.
   Przy przeładowaniu ważna jest tylko odmienność listy argumentów. Typ zwracany przez funkcję nie jest brany pod uwagę.

   - Zatem takie przeładowanie:

     ```
     void fun (int);
     void fun (double);
     void fun (int, int);
     void fun (int, double);
     ```

   jest poprawne.

   - Natomiast takie:

     ```
     int funkcja (int);
     double funkcja (int);
     ```

   jest niepoprawne.

2. **Funkcja inline**
   - ang. in line, czyli w linii - kompilator umieszcza ciało funkcji w linijce, w której jej wywołanie następuje; zatem jeżeli funkcję zamierzamy wywoływać tysiące razy to czas zużyty na wywołanie i powrót z funkcji, wtedy sens ma tworzenie w funkcji „w linii".
   - Funkcje inline zasadniczo są przewidziane dla małych i krótkich funkcji (zwykle maksymalnie dwulinijkowych).
   - Jeżeli definicję funkcji składowej umieścimy bezpośrednio w ciele klasy, to taka funkcja będzie automatycznie traktowana jak funkcja typu inline.
     Oczywiście w praktyce ma to sens jeżeli funkcja jest funkcją krótką.

3. **Konstruktor:**
   - Specjalna funkcja składowa, która nazywa się tak samo jak klasa.
   - Konstruktor może być przeładowany.
   - C++11:

```
class T_Klasa
{
  private:

       int skladnik1, skladnik2;

   public:
      T_Klasa(int n); //konstruktor jednoargumentowy napisany przez programistę
      T_Klasa() = default; //"prośba" do kompilatora o utworzenie domniemanego

                                  //konstruktora mimo, że w programie jest inny konstruktor


};
```

```
class T_Klasa2
{
  private:

       int skladnik1, skladnik2;

   public:
       //brak jakiegokolwiek konstruktora
       T_Klasa() = delete; //”prośba” do kompilatora o nie tworzenie domniemanego

                                  //konstruktora mimo, że w programie NIE ma żadnego
                                   //konstruktora


};
```