Základy programování v C++ 23. cvičení Šablony

Zuzana Petříčková

15. prosince 2019

Přehled

Šablony v C++

Šablony (templates)

- vzor, podle kterého vytváří překladač různé podobné funkce nebo objektové typy (tzv. **instance šablony**)
- typy šablon:
  - šablony funkcí
  - šablony tříd

Deklarace šablony:

```
template<seznam_parametru> deklarace
```

- parametry (oddělené čárkou): typové, hodnotové, šablonové
- klíčová slova: **template, typename**

```
template<typename T>
T max(T a, T b)
{
    if (a > b)
      return a;
    else
      return b;
}
```

Šablony funkcí ... příklad 1

```
template<typename T>
void prohod(T &a, T &b)
{
    T c = a;
    a = b;
    b = c;
}
int main()
{
   int x = 2, y = 3, z = 0;
   prohod(x,y);
   prohod<int>(y,z);
   ...
}
```

- **T** je libovolný datový typ

Šablony funkcí ... příklad 2

```
template<typename T, int n>
void prohod(T *a, T *b)
{
    T p;
    for (int i = 0; i < n; i++)
    {
        p = a[i];
        a[i] = b[i];
        b[i] = p;
    }
}
int main()
{
   int x[] = {3,4,5,6}, y[] = {0,1,2}, z[] = {8,7};
   prohod<int,3>(x,y);
   prohod<int,2>(y,z);
}
```

- šablona má jeden typový parametr a jeden hodnotový parametr (při volání to musí být konstanta)

Šablony tříd ... příklad 1

```
template<typename T>
struct Uloziste
{
    T data;
};

int main()
{
    Uloziste<int> bedna;
    bedna.data = 3;

    Uloziste<string> b1;
    b1.data = "svetr";

    Uloziste<double>* b2 = new Uloziste<double>;
    b2->data = 3.14;
    delete b2;
    ...
}
```

Šablony tříd ... příklad 2

```
template<typename T>
class Uloziste1
{
    T data;
public:
    Uloziste1(T _data) : data(_data) { }
    void nastavData(T data)          { this->data = data; }
    T vratData()                     { return data; }
};
int main()
{
    Uloziste1<int> bedna(9);
    cout << bedna.vratData() << endl;
    bedna.nastavData(3);
    cout << bedna.vratData() << endl;

    Uloziste1<string>* bedna1 = new Uloziste1<string>("svetr");
    cout << bedna1->vratData() << endl;
    delete bedna1;
    ...
}
```

Šablony ... organizace programu

- pro šablony fukcí a tříd nelze použít běžné dělení programu do hlavičkových a zdrojových souborů

Dvě aktuálně používané strategie:

1. celá definice šablony je v hlavičkovém souboru
   - informativní i definiční deklarace jsou přímo v hlavičkovém souboru
2. v hlavičkovém souboru jsou pouze informativní deklarace
   - pak musíme instance šablon vytvořit explicitně:

```
// sablony.h
template<typename T> T vrat(T a);

// sablony.cpp
template<typename T>
T vrat(T a)
{
    return a;
}
template int vrat(int);
template float vrat(float);
```

Šablony ... explicitní generování instancí

```
template deklarace;


template class Uloziste1<int>;
template class Uloziste1<string>;

template int max(int,int);
template float max(float,float);
```

Šablony tříd ... příklad 3

```
template<typename T>
class Uloziste2
{
        T data;
public:
        Uloziste2(T _data);
        void nastavData(T data);
        T vratData();
};

template<typename T>
Uloziste2<T>::Uloziste2(T _data) : data(_data)
{
}
template<typename T>
void Uloziste2<T>::nastavData(T data)
{
        this->data = data;
}
...
```

- Informativní a definiční deklarace metod v jednom (hlavičkovém) souboru

Příklad na procvičení I: Chytré pole

- Přeměňte třídu **ChytrePole** na šablonu třídy tak, aby v chytrém poli mohly být uloženy hodnoty libovolného typu.
  - deklaraci šablony i definiční deklarace metod šablony vložte do souboru ChytrePole.h

Příklad na procvičení II : Spojový seznam

Na rozmyšlenou (dobrovolná domácí úloha)

- Přeměňte struktury **Prvek** a **Seznam** ze cvičení na šablony tříd tak, aby v seznamu mohly být uloženy hodnoty libovolného typu.