Základy programování v C++ 24. cvičení
Vstupní parametry programu
Podmíněný výraz
Ukazatele na funkce

Zuzana Petříčková

9. ledna 2020

Přehled

O čem jsme nehovořili

Náměty k samostudiu

- datový typ Union (obdoba struktur)
- aserce (alternativní ošetření chyb za běhu)
- více o objektově orientovaném programování: statické atributy a metody, (virtuální) dědění, polymorfismus: bude probíráno v druhém ročníku

Dnes

- vstupní parametry programu
- podmíněný výraz
- ukazatele na funkce
- šablony funkcí a tříd

Vstupní parametry programu

Funkce main()

- musí být typu **int**
- nelze ji volat rekurzivně
- může mít až dva parametry přesně určených typů (některé implementace dovolují i třetí parametr)

Parametry funkce main()

```
int main(int argc, char* argv[]);
```

- typy parametrů jsou povinné, jména volitelná
  - **argc** ... počet parametrů na příkazové řádce
  - **argv** ... pole řetězců (parametry na příkazové řádce)

Vstupní parametry programu ... příklad

Parametry funkce main()

```
int main(int argc, char** argv)
{
    cout << "Vstupni parametry programu:" << endl;

    for (int i = 0; i < argc; ++i)
        cout << argv[i] << "\n";

    return 0;
}
```

- Nastavení parametrů funkce main() ve Visual Studiu:
  - ,,right click on the project name, then go to Properties. In the Properties Pane, go to ,,Debugging", and in this pane is a line for ,,Command-line arguments."

Podmíněný výraz

- syntaxe: **E ? E1 : E2**
- je-li hodnota výrazu **E** rovna **true**, je výsledkem podmíněného výrazu **E1**, jinak **E2**.

Příklad

```
double x, y, m;
cin >> x >> y;
...
x = (x >= 0)  ? x : -x; // absolutni hodnota
m = (x >  y)  ? x : y;  // maximum dvou cisel

double a = 0, b = 1;
bool v = (x >= a) ? (x <= b) ? true : false : false;
```

Ukazatele na funkce

V jazycích C/C++ jsou dva typy ukazatelů:

- ukazatele na data (už známe)
- ukazatele na funkce
  - obsahují adresu vstupního bodu do funkce

K čemu je dobrý ukazatel na funkci?

- funkce jako proměnná / parametr jiné funkce
- pole ukazatelů na funkce (programování nabídky/menu)

Ukazatel na funkci

Deklarace ukazatele na funkci:

```
typ (*identifikator)(seznam_typu_parametru);
```

Příklady

```
int (*F) (int, int);
    /* F je ukazatel na funkci typu int
       se dvema parametry typu int */

void (*G) (Zamestnanec*, double);
    /* G je ukazatel na funkci bez navratove hodnoty
       se dvema parametry typu ukazatel na Zamestnance a double */

bool (*H) (void);
    /* H je ukazatel na funkci typu bool bez parametru */

Prvek* (*I) (Seznam&);
    /* I je ukazaten na funkci, ktera vraci ukazatel na Prvek
       s parametrem typu reference na seznam */
```

Ukazatel na funkci

Přiřazení hodnoty (v C++ dvě ekvivalentní možnosti):

```
double (*S) (double);
S = cos;
S = &cos;          // jazyk C
```

Volání funkce, na kterou ukazatel ukazuje
(v C++ dvě ekvivalentní možnosti)

```
double x;
x = S(3.14);
x = (*S)(3.14);      // jazyk C
```

Ukazatel na funkci ... příklad

Rostoucí funkce

```
bool rostouci(double (*f)(double),double a,double b,double krok)
{
    for (double x = a; x+krok<=b; x+=krok)
    {
        if(f(x)>=f(x+krok))
          return false;
    }
    return true;
}
int main()
{
    if (rostouci(sin,-pi/2,0,0.0001))
        cout << "sin roste" << endl;
    if (rostouci(cos,-pi/2,0,0.0001))
        cout << "cos roste" << endl;
     ...
}
```

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)
{
    return (a > b) ? a : b;
}
```

Šablony funkcí ... příklad

```
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 tříd ... příklad

```
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;
    ...
}
```

Příklady

1. Napište a zavolejte funkci, která pomocí podmíněného výrazu spočte hodnotu (pro $a < b, a, b \in R$):

$$f(x) = \begin{cases} a, & x < a \\ x, & a \leq x \leq b \\ b, & x > b \end{cases}$$

2. Napište šablonu funkce, která otočí pořadí prvků v poli délky n.
3. Napište funkci, která se pokusí nalézt kořeny rovnice na nějakém intervalu (obdoba funkce rostouci).

```
double koreny(double(*F)(double),
              double a, double b, double eps);
```

pomocí funkcí spočtěte kořeny následujících funkcí:

- $(tg(x) - x)$ na intervalu $(\pi/2, 3\pi/2)$
- $(\sqrt{x+2} - 2)$ na intervalu $(-2, 5)$