Základy programování v C++ 12. cvičení Textové řetězce

Zuzana Petříčková

31. října 2019

Přehled

Textové řetězce v C/C++

Textové řetězce v jazyce C

```
char retezec[delka];
```

- v jazyce C reprezentujeme textový řetězec jako pole znaků
- s textovým řetězcem pak zacházíme stejně jako s libovolným polem
- **výhoda**: efektivní práce (prostorová i časová úspora)
- funkce v hlavičkovém souboru **string.h (cstring)**

Textové řetězce v C++

```
string retezec;
```

- navíc spec. objektové datové typy pro reprezentaci řetězců (**string**, **wstring**)
- **výhoda**: jednodušší a bezpečnější práce (chová se podobně jako struktura), řada metod a operátorů
- funkce v hlavičkovém souboru **string**

Textové řetězce v C (i C++)

```
char retezec[delka];
```

- v jazyce C má význam textového řetězce pole znaků
- textový řetězec musí být ukončený znakem '\0'
  → pole musí být alespon o 1 delší než je uložený řetězec

Inicializace

```
char s[10] = "text"; //prekladac automaticky doplni '\0'
char s1[10] = {'t','e','x','t','\0'};
```

- s textovým řetězcem se pak zachází stejně jako s libovolným polem
- Funkce pro práci s textovými řetězci v hlavičkovém souboru **string.h (cstring)**

Textové řetězce v C (i C++)

Práce s textovými řetězci

- přístup k prvku pole pomocí indexovacího operátoru []
- tisk na konzoli pomocí **cout**
- načtení z konzole: **cin**, **cin.getline()**

```
char s[256], t[256], u[256];
cin >> s;  // nacte text po prvni bily znak (' ','\t','\n')
cin.getline(t, 255); // nacte text do konce radku (znak '\n
cin.getline(u, 255, ';'); //nacte text az po znak ';'

cout << s << endl << t << endl u << endl;

// t = s          // NELZE
// t = "pes";     // NELZE
// if (t == s)    // NELZE

s[0] = '\0';      // vyprazdnim retezec
cout << s << endl;
```

Textové řetězce v C (i C++)

Příklady (ukázka)

- Vlastní implementace funkce **int delkaRetezce(const char s[])**, která spočítá délku řetězce.
- Vlastní implementace funkce **void kopirujRetezec(char t[], const char s[])**, která zkopiruje obsah řetězce **s** do řetězce **t**.

Textové řetězce v C/C++

Příklad: délka řetězce

```
int delkaRetezce(const char s[])
{
    int n = 0;
    while (s[n] != '\0')
        n++;
    return n;
}
```

- knihovní funkce **strlen()**

```
char s1[10] = "text";
cout << delkaRetezce(s1) << endl;
cout << strlen(s1) << endl;
```

Textové řetězce v C/C++

Příklad: kopírování řetězců

```
void kopirujRetezec(char t[], const char s[])
{
    int i = 0;
    do {
        t[i] = s[i];
        i++;
    } while (s[i-1] != '\0');
}
```

- knihovní funkce **strcpy_s(kam,pocet_bytu,odkud)**

```
char s[10] = "text", t[10], u[10];
kopirujRetezce(t, s);
strcpy_s(u,sizeof(s),s);
cout s << endl << t << endl << u << endl;

// lexikograficke porovnani
if (strcmp(t, u) == 0)
  cout << "t a u jsou stejne.";
```

Textové řetězce v C/C++

Velikost datového typu v bytech

- funkce **sizeof()**

```
...
const int n = 3;
int a[n];
int x;

int velPole = sizeof(a);
int velInt = sizeof(int);
int velX = sizeof(x);
int pocetPrvku = velPole/velInt;
...
```

Délka statického pole

Funkce **sizeof()** musí "bezprostředně následovat" po definici pole:

```
...
const int n = 10;
char a[n] = "muj text";

int velPole = sizeof(a); // velikost pole v bytech
int velChar = sizeof(char); // velikost char-u v bytech
int pocetPrvku = velPole/velChar;
cout << "Pocet prvku pole je" << pocetPrvku << endl;
cout << "Delka textu je" << strlen(a) << endl;
...
```

Funkce **sizeof()** volaná ve funkci nedá správný výsledek:

```
void vypisDelku(const char a[])
{
    cout << "Delka textu je" << strlen(a) << endl;
    cout << "Delka pole NENI " << sizeof(a)/sizeof(char);
}
```

Funkce pro práci s textovými řetězci

Výběr funkcí z knihovny string.h (cstring) :

strlen()	délka řetězce
strcpy_s()	kopírování obsahu řetězce z pole do pole
strncpy_s()	kopírování řetězce délky n
strchr()	vyhledání prvního výskytu znaku
strrchr()	vyhledání posledního výskytu znaku
strstr()	vyhledání prvního výskytu podřetězce
strcmp()	lexikografické porovnání řetězců
strcat()	připojení řetězce k řetězci

...

Textové řetězce v C++

- navíc objektové datové typy pro reprezentaci řetězců:
  - **string** ... řetězce z úzkých znaků
  - **wstring** ... řetězce z Unicode znaků
- knihovna funkcí ... hlavičkový soubor **string**
- Výhody:
  - jednodušší a bezpečnější práce (chová se podobně jako struktura)
  - řada metod a operátorů:
    - = pro přiřazení
    - [] pro indexaci
    - + a+= pro zřetězení
    - <,<=,>,>=,==,!= pro lexikografické porovnání

Textové řetězce v C++

Datový typ string

```
string s1 = "text", s2 = { 't','e','x','t','\0' };
string s3, s4, s5;

int delka = s1.length();

// nacitani a vypis
cin >> s3;
getline(cin, s4);
getline(cin, s5, '.');
cout << s3 << endl;

// indexace
cout << s1[0] << endl;
cout << s1.at(0) << endl;
```

Textové řetězce v C/C++

Příklad: délka řetězce
(úplně stejně jako pro pole znaků)

```
// int delkaRetezce(string s)
int delkaRetezce(const string& s)
{
    int n = 0;
    while (s[n] != '\0')
        n++;
    return n;
}

    string s = "text";
    cout << delkaRetezce(s) << endl;
    cout << s.length() << endl;
```

Textové řetězce v C++

Datový typ string

```
string s1 = "text", s2 = { 't','e','x','t','\0' };
string s3, s4, s5;
...
s5 += s3 + " " + s4;        // konkatenace (zretezeni)
s3 = s1;                    // kopirovani
s3 = "jiny text";           // kopirovani
if (s3 < s4)                // porovnani
    cout << "je lexikograficky mensi" << endl;
if (s3 == s4)               // porovnani
    cout << "retezce jsou identicke" << endl;
char x[20];
kopirujRetezec(x,s3.data());  // prevod na pole charu
kopirujRetezec(x,s3.c_str()); // prevod na pole charu
...
```

Textové řetězce v C/C++

Už jsme měli

- Funkce **char naVelke(char c)**, která převede malé písmeno na velké. Pokud znak není malé písmeno, vrátí ho beze změny.
- Funkce **void celyRadek()**, která postupně načte (znak po znaku) z konzole celý řádek (až po znak '\n') a vypíše ho na konzoli s tím, že všechna malá písmenka změní na velká.

Textové řetězce v C/C++

Už jsme měli

```
char naVelke(char znak)
{
    if (znak >= 'a' && znak <= 'z')
        znak = znak - 'a' + 'A';
    return znak;
}

void celyRadek()
{
    cout << "Zadej radek textu:" << endl;
    char znak;
    do {
        znak = getchar();
        cout << naVelke(znak);
    } while (znak != '\n');
}
```

Textové řetězce v C/C++

Příklady

- Napište a zavolejte funkci **string naVelka(string s)**, která změní všechna malá písmena v řetězci na velká.
- Napište a zavolejte funkci **string zasifrujSubstituci(string zprava, int posun)**, která zašifruje text **zprava** pomocí substituční (Cézarovy) šifry (posun velkých písmen o **posun** znaků (jiné znaky zachová šifra beze změny)).

Alternativně (pomocí referencí):

- **void naVelka(string &s)**
- **void zasifrujSubstituci(string &zprava, int posun)**

Textové řetězce v C/C++

Starší příklad (pomocí řetězců)

- Funkce **void celyRadek()**, která postupně načte (znak po znaku) z konzole celý řádek (až po znak '\n') a vypíše ho na konzoli s tím, že všechna malá písmenka změní na velká.

```
void celyRadek()
{
    cout << "Zadej radek textu:" << endl;
    string s;
    getline(cin, s);
    cout << naVelka(s) << endl;
}
```

Textové řetězce v C/C++

Další příklady na procvičení (různá obtížnost)

- Napište vlastní funkci **int doDesitkove(const string &s, int zaklad)**, která převede číslo v soustavě o základu **zaklad** zadané jako textový řetězec na číslo v desítkové soustavě (analogie knihovní funkce **stoi()**).
- Napište funkci **bool jePalindrom(const string &text)**, která vrátí true, pokud je text palindromem (tj. čte se stejně popředu jako pozpátku, př. "kobylamamalybok").
- Napište funkci **string zasifrujSubstituci(string zprava, const string &vzor)**, která zašifruje text **s** pomocí substituce zadané řetězcem **vzor**. (např. z = zasifrujSubstituci("ABECEDA 1", "KLMN"); z bude rovno "KLEMENK 1")