**Algoritmi un programmēšana.   
Lekcija 7. Simboli un simbolu virknes (konspekts).**

DatZ1165: Algoritmi un programmēšana  
kursa nodarbību plāns 2020R

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 7. | 2020.10.20. | Simboli. C++ zema līmeņa simbolu virknes char\*.  Augsta līmeņa simbolu virknes string valodā C++. Simbolu virknes valodā Python. Norādes, adreses un references. Masīvi un norādes. | US |  | [ZC] 9, 12  [ZP] 11 | [HC] 3.8, 10.7, 10.8, 11.2.6, 13.1  [HP] 2.1, 2.5, 2.6 |

Studēsim 7.lekcijā paredzētos tematus.

Pieeja studēšanai būs 6K (angliski 6W) – sešu aspektu aplūkošana:

Kas (Who),

Kur (Where),

Kad(When),

Ko (What),

Kā (HoW),

Kāpēc (Why).

6K lieto izmeklēšanā, žurnālistikā, zinātnē utt.

**Svarīgi – MILZĪGA atšķirība pieejā datiem C++ valodā no Python valodas:  
C++ ir svarīga atmiņas organizācija,  
Python tikai vērtības ir svarīgas.**

Vielas apgūšanai studēsim ieteiktos materiālus (un dažus citus).

[ZC] ir elektroniskā grāmata Jānis.Zuters. Programmēšana un C++, Latvijas Universitāte, 2007-2020, 228 lpp.

[ZP] ir elektroniskā grāmata Jānis Zuters. Programmēšanas pamati ar valodu Python, Latvijas Universitāte, 2019-2020, 162 lpp.

**Simbols.**

simbols  
Rakstzīme, ko izmanto datu grafiskai izteikšanai vai kodēšanai datorā ([www.letonika.lv](http://www.letonika.lv)).

Svarīgākās ziņas par simbola lietošanu C++.

No informācijas apstrādes viedokļa tips char ir tas pats int, tikai vienā baitā, jo atmiņā simbolus reprezentē to kodi. ASCII kodu tabula ir standartizēta visā pasaulē.

Veidot: char c = 'A';

Mainīt: c = ‘7’;

Ievadīt: cin >>c; cin.get(c); /// Var patvaļīgu simbolu ievadīt

Izvadīt: cout << c;

Lai noskaidrotu, ka simbols mainīgajā c ir cipars, der nosacījums:

c>='0' && c<='9'

vai

c>=48 && c<=57

Standarta bibliotēka <cctype> piedāvā funkciju klāstu simbolu apstrādei.

isalpha

int isalpha (int ch);

Funkcija isalpha() atgriež no nulles atšķirīgu vērtību, ja padotais arguments ch ir burts (latīņu lielais vai mazais).

isalpha('A') // atgriež 1

**Simbolu virkne.**

Simbolu virkne (string) ir simboliskā tipa (piemēram, char) vērtību secība, kas reprezentē tekstuālu informāciju.

Valodā C++ ir pieejamas divu veidu simbolu virknes:

zema līmeņa (C stila) simbolu virknes (char\*),

augsta līmeņa (C++ stila) simbolu virknes (string).

**Zema līmeņa simbolu virknes.**

Zema līmeņa (C stila) simbolu virkne ir simboliskā tipa vērtību secība, kuru noslēdz speciāls simbolu virknes beigu simbols.

Veidot: char s1[15] = "Hallo, world!";

Mainīt: s1[1] = ‘e’;

Ievadīt: cin >>s1; cin.getline (s1, 15); /// Var patvaļīgu simbolu virkni ievadīt

Izvadīt: cout << s1;

Standarta bibliotēka <string.h> piedāvā funkciju klāstu zema līmeņa simbolu virkņu apstrādei.

strcmp

int strcmp (const char\* s1, const char\* s2);

Funkcija strcmp() leksikogrāfiski salīdzina virknes s1 un s2, atgriežot veselu skaitli:

0, ja virknes vienādas;

<0, ja s1 mazāka par s2;

>0, ja s1 lielāka par s2.

strlen

size\_t strlen (const char\* s);

Funkcija strlen() atgriež simbolu virknes i garumu.

char t[20]={'H','e','l','l','o',0};

cout << t << setw(3) << strlen(t) << endl; /// Hello 5

Saturīgs piemērs failā **find\_Cstring.cpp.**

**Augsta līmeņa simbolu virknes.**

Augsta līmeņa simbolu virknes valodā C++ ir objekti (lasi, mainīgie), kas nodrošina teksta glabāšanu un apstrādi, rūpējoties par nepieciešamo atmiņas vietu virknes saglabāšanai un

nodrošinot ērtu sintaksi dažādām darbībām.

Augsta līmeņa simbolu virkne izmanto bibliotēku <string>.

Veidot: string s = "Hello, world";

Mainīt: s[1] = ‘a’; s += “!!!”; /// Hallo, world!!!

Ievadīt: cin >>s; getline (cin, s); /// Var patvaļīgu simbolu virkni ievadīt

Izvadīt: cout << s;

Augsta līmeņa simbolu virknes garumu var noskaidrot, izmantojot simbolu virknes objekta funkcijas size() un length() (strādā identiski).

cout << s.size() << endl;

Saturīgs piemērs failā **find\_string.cpp.**

**Norādes.**

Norāde (pointer) ir mainīgais (vai cits mehānisms), kas glabā

(nosaka) atmiņas adresi.

Veidot: char \*arr2;

Mainīt: char arr[20] = “LU is OK”; arr2 = arr;

Ievadīt: nevar

Izvadīt: cout << arr2;

Var veikt aritmētiskas darbības - arr2 += 3; // pārvieto norādi 3 vienības uz priekšu.

Saistībā ar norādēm izšķir 2 pretējas darbības:

- adreses iegūšana noteiktam atmiņas apgabalam pēc to reprezentējošā mainīgā,izmantojot adreses ņemšanas operatoru & (nesajaukt ar referenci!),

- vērtības iegūšana pēc adreses (nesajaukt ar norādes mainīgā deklarēšanu!).

Vērtības iegūšana pēc adreses notiek, pirms norādes liekot zvaigznītes operatoru \*:

int arr[4] = {33,88,11,66};

int \*p; // norādes mainīgā deklarēšana

p = &arr[2]; // arr[2] adreses piešķiršana mainīgajam p

cout << \*p << endl; // satura (int) izdruka pēc adreses p /// 11

**Python simbolu virknes.**

Simbolu virknes (string) ir simbolu secības, kas nodrošinātas ar dažādām funkcijām ērtākai teksta apstrādei.

Šī mācību materiāla ietvaros apskatīsim parasto (Unicode) simbolu virkni (datu tips: str), kas automātiski pieņem un glabā simbolus jebkurā kodējumā (valodā), un šīs visas ērtības nodrošināšanai Python var tērēt pat daudzus baitus uz katru simbolu.

Simbolu virkne str ir nemaināmais (immutable) datu tips.

Veidot: s = 'Hello'

Mainīt: nevar. Tiek veidots jauns objekts ar tādu pat vārdu:

s += ', World!' # tagad s jau ir cita virkne, nevis pamainīta vecā

Ievadīt: s = input(“Ievadiet virkni”)

Izvadīt: print(s) print(s[2])

Derīgi pārveidojumi:

Intervāla operators : (slice) –

s = 'DATORIKA'

print(s[2:5]) # [a:b] - no a līdz b neieskaitot

print(s[:5]) # ja a neuzrāda, tad no sākuma

print(s[2:]) # ja b neuzrāda, tad līdz beigām

print(s[:]) # visas virknes izdrukāšana, pirms tam izveidojot tā kopiju

Virknes elementa piederības pārbaude –

s = 'HELLO'

print('L' in s) # True

Saturīgs piemērs failā **PigLatin\_word.py**

Virkņu formatēšanas uzlabojums f-String (sākot no Python 3.6) - https://realpython.com/python-f-strings/

Simbolu virkni saskalda apakšvirknēs ar split (apakšvirkņu robežas nosaka tukšuma simboli), un izveidojas saraksts no apakšvirknēm –

s = ' a b c '

ss = s.split() # saskalda pēc tukšumiem

print(ss) # ['a', 'b', 'c']

Pretēja darbība notiek ar join, kas izveido simbolu virkni no saraksta ar virknēm.

Varētu papildināt saturīgo piemēru PigLatin\_word.py, lai apstrādātu nevis vienu vārdu, bet teikumu no vairākiem vārdiem.