Stringovi

U C programskom jeziku ne postoji tip string, nego se koristi niz znakova (niz čiji je osnovni tip char). Kod takvih "C stringova" moramo voditi računa oko puno detalja, npr. ako želimo stringu dodati neke znakove moramo voditi računa ima li dosta mjesta u nizu za te znakove. U C++ se također može raditi sa "C stringovima" jer je programski jezik C podskup jezika C++, ali da bi se programerima olakšao život, u C++ je uveden tip string koji se ponaša dosta predvidivo.
Ova klasa je smještena u biblioteku <string> a sve definicije su smještene u namespace std. Za naš kompajler pri radu sa stringovima direktiva #include <string> nije potrebna, ali je uključite zbog portabilnosti vašeg koda. Evo kako izgleda rad sa stringovima:

```
string  s1, s2, s3;
s1=''auto'';
s2=''parking'';
s3=s1+s2;
cout<<"Ja trebam "+s3;
```

Kao što smo vidjeli i ranije, string se stavlja u dvostruke znake navoda. Ukoliko želimo dva stringa spojiti u treći, onda možemo koristiti operator +. Operator = je, kao i obično, za dodjelu vrijednosti. Dakle stvari se ponašaju kako i pretpostavljamo, s tim da treba napomenuti da je "auto" ovdje C string i da se on ovdje automatski pretvara u tip string kada se sprema u s1.

Kod zadnje izjave miješaju se C i C++ stringovi pa kompajler mora raditi dosta da ovdje sve stvari funkcionišu kako treba.
Kada želimo unositi string sa tastature onda treba voditi računa da cin ignoriše blanko znakove, npr.

```
string s1,s2;
cin>>s1;
cin>>s2;
```

Ovdje ako korisnik unese rečenicu:"Ja idem u šetnju.", s1 će imati vrijednost "Ja" a s2 "idem".

```
string linijaTeksta;
getline(cin,linijaTeksta);
```

Ako želite unijeti cijelu rečenicu u jedan string, onda morate koristiti funkciju **getline**.

Dakle ovakav način učitava sve do pojave znaka '\n', tj. pritiska tipke enter. Postoje i druge verzije funkcije getline, recimo ako želimo stati sa učitavanjem linije kada neko unese znak ?, onda bi išli sa:

```
getline(cin,linijaTeksta, '?');   // ne učitava  znak ?
```

Upozorenje: Treba paziti prilikom miješanja funkcija cin i getline da ne dobijete neželjene rezultate:

```
int n;
string linija;
cin>>n;
getline(cin,linija);
```

Ako korisnik unese:
42
Zdravo svima!
očekujemo da vrijednost varijable n bude 42 a stringa linija "Zdravo svima!".

Ipak nije tako jer cin kod unosa broja odbacuje sve nevidljive znakove pa tako i enter nakon unosa broja, što čeka narednu liniju getline za učitavanje, a ta komanda će odmah stati sa učitavanjem na znaku enter, pa će string linija biti prazan!!! Da bi se ovaj problem riješio možete koristiti cin.ignore(100, '\n');, što je poziv ignore funkciji koja će čitati i odbacivati cijeli ostatak linije(odbačen od cin iz prethodne linije) zaključno sa i uključujući i znak '\n'(ili ako ne nađe '\n', onda dok ne odbaci 100 znakova). Međutim, jednostavniji način je upotrijebiti objekat nazvan "gutač praznina" (engl. whitespace eater ili whitespace extractor), koji je u jeziku C++ imenovan prosto imenom "ws". Radi se o specijalnom objektu za rad sa tokovima (poput objekta "endl"), koji je također neka vrsta manipulatora. Ovaj objekat, upotrebljen kao drugi argument operatora izdvajanja ">>", uklanja sve praznine koje se eventualno nalaze na početku ulaznog toka, ostavljajući sve druge znakove u ulaznom toku netaknute.
Jednostavno uradite ovo:

```
cin>>n>>ws;
```

Ukoliko želite i na ove C++ stringove gledati kao na niz znakova, onda i to možemo, što se vidi iz:

`duzina=linijaTeksta.length();`
`for (int i=duzina-1;i>=0;i--)`
`cout<<linijaTeksta[i];`
	Ovdje se uz pomoć funkcije length, određuje dužina stringa, koji se onda štampa od zadnjeg do prvog znaka, tj. obrnuto. To se može raditi i sa at()	`duzina=linijaTeksta.length();`
		`for (int i=duzina-1;i>=0;i--)`
		`cout<<linijaTeksta.at(i));`

Dakle oba ova načina za pristup pojedinim znakovima stringa su dozvoljena.
Operatori == i = rade na predvidiv način i sa stringovima. Operatori <, >,<= i >=, uspoređuju stringove leksikografski, tj. po abecedi, a ne po dužini stringa. Tako bi npr. string "apple" bio "manji" od stringa "banana", iako je duži od drugog stringa. Evo nekih funkcija sa stringovima:

`str.substr(pos, length)`	vraća podstring stringa str od mjesta pos i dužine length
`str.empty()`	vraća tačno ako string str prazan
`str.insert(pos,str1)`	umeće string str1 u string str počevši od pozicije pos u stringu str
`str.erase(pos, length)`	uklanja iz stringa str podstring dužine length, počevši od mjesta pos
`str.find(str1)`	pronalazi poziciju prvog pojavljivanje stringa str1 unutar str. Ako ne nađe vraća broj -1.
`str.find(str1, pos)`	pronalazi poziciju prvog pojavljivanje stringa str1 unutar str počevši od pozicije pos.
`str.c_str()`	vraća c string sa istim znakovima kao i str

Napiši program koji okreće riječ naopako!

```
#include <iostream>
#include <string>
using namespace
std;
int main ()
{
  string moj_string;
  int duzina;

  getline(cin, moj_string);
  duzina=moj_string.length();

  for(int i=duzina-1;i>=0;i--)
    cout<<moj_string[i];

  system("Pause");
  return 0;
}
```

Napiši program koji provjerava da li je riječ palindrom

```
#include <iostream>
#include <string>
using namespace std;

int main ()
{
  string ulaz, zalu;
  int duzina;
  getline(cin, ulaz);
  duzina=ulaz.length();
  zalu="";
  for(int i=duzina-1;i>=0;i--)
    zalu=zalu+ulaz[i];
  if (ulaz==zalu)
    cout<<"Unesena rijec je palindrom";
  else
    cout<<"Unesena rijec nije palindrom";
  system("Pause");
  return 0;
}
```

1) Napišite program koji od korisnika traži dvije riječi,a zatim provjerava da li je druga riječ dio prve.

Primjer1
Unesi prvu rijec:buna
Unesi drugu rijec:una
Druga rijec je dio prve.

Primjer2
Unesi prvu rijec:bosna
Unesi drugu rijec:posna
Druga rijec nije dio prve.

2) Napišite program koji provjerava da li je druga riječ prefiks ili sufiks prve riječi.

Primjer1
Unesi prvu rijec:bunar
Unesi drugu rijec:nar
Jeste prefiks/sufiks.

Primjer2
Unesi prvu rijec:bunar
Unesi drugu rijec:buna
Jeste prefiks/sufiks.

Primjer3
Unesi prvu rijec:bunar
Unesi drugu rijec:una
Nije prefiks/sufiks.