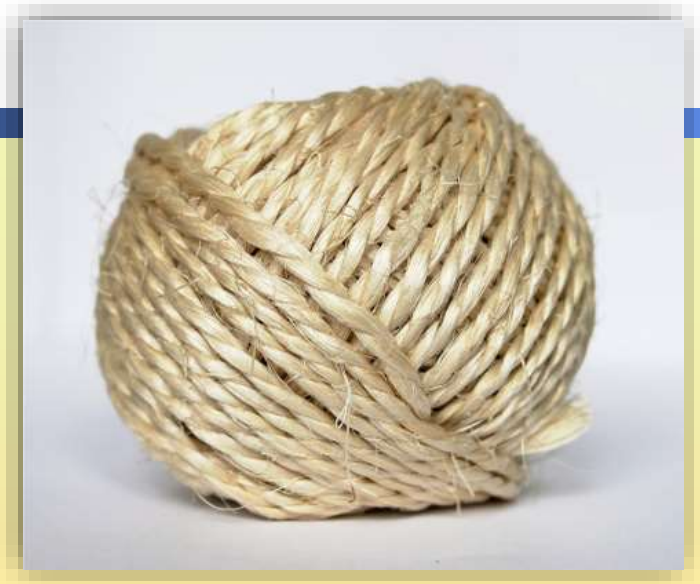
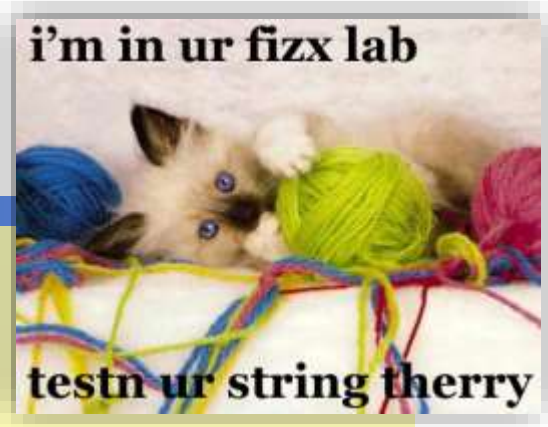


C++ razred `string`

Upotreba C++ razreda (klase) `string` u programima

Uvod

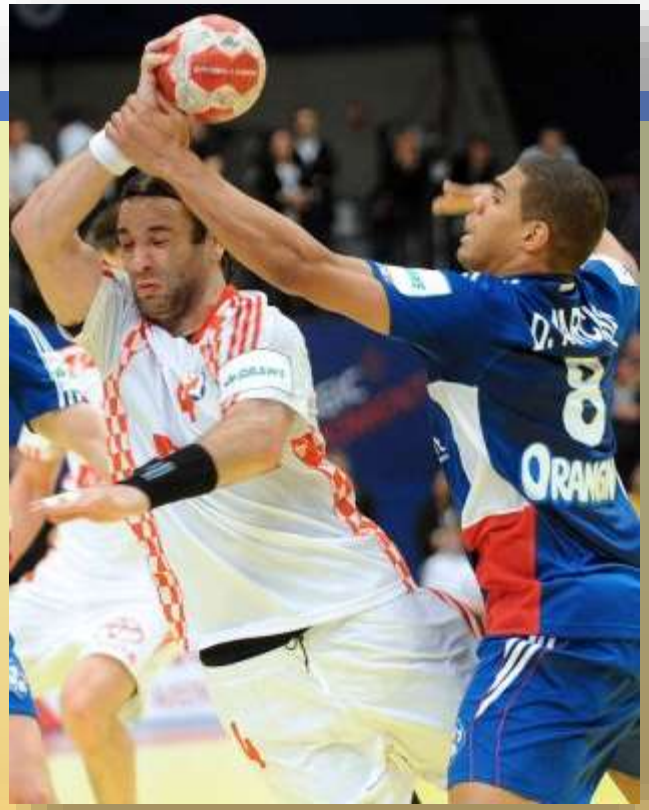


- relativno novi dodatak za C++
- kao alternativa upotrebi znakovnih polja i C-stringova za manipuliranje tekstom može se koristiti objekte iz C++ razreda `string`
- objekti iz razreda `string` su superiorniji:
  - ne trebamo postavljati određenu veličinu polja (koja bi se mogla slučajno prekoračiti)
  - preporuča se upotrebljavati u novim programima
  - poboljšava sposobnost programera da rukuje tekstom na uobičajen i siguran način
  - lakše se vrše promjene i manje je grešaka

C++ razred `string` - prednosti

- prednosti upotrebe objekta iz razreda `string`:
  - C++ **automatski vodi brigu o veličini teksta** koji je u njemu, **rezervira dovoljno memorije** i prema potrebi **automatski proširuje memorijsko područje** koje se koristi za čuvanje teksta
  - ne moramo specificirati veličinu `string` objekta kao što moramo kad deklariramo znakovno polje
  - manja je mogućnost prepisivanja preko susjednih memorijskih ćelija jer C++ koristi određeni *range checking*
  - Možemo koristiti **operatore** za izvođenje nekih manipulacija stringovima, npr. = za pridruživanje, a + za spajanje stringova

C++ razred `string` – primjer 1.



```
#include <iostream>
#include <string>

using namespace std;

int main() {
   string s1, s2, s3;
   s1 = "Rukometna ";
   s2 = "reprezentacija.";
   s3 = s1 + s2;
   if (s1<s2) cout << s3 << endl;
        //leksikografska usporedba dva stringa
   return 0;
}
```

Deklariranje, inicijalizacija, ispis, operatori

- null znak nije nužno posljednji u tekstu objekta `string`. Tipično, određeni podatkovni član razreda `string` čuva veličinu teksta objekta i taj se podatkovni član koristi u manipulacijama stringom.
- C++ ima preopterećene (prilagođene) neke aritmetičke i relacijske operatore da rade sa `string` objektima (+, <)
- **Leksikografska usporedba**
  - Ako su samo mala ili samo velika slova u stringovima, svodi se na alfabetsku usporedbu: jedan je string manji od drugog ako dolazi prvi po abecedi. Ako su slova pomiješana ili ima simbola, onda se usporedba bazira na cjelobrojnom ekvivalentu svakog znaka (prema ASCII znakovnom skupu velika slova imaju cjelobrojne ekvivalente manje od malih slova).

Operatori za `string` objekte

Pridruživanje	=	Sprema string u objekt string
	+=	Spaja i sprema string u string objekt
Leksikog usporedba	==	Istina ako su dva stringa identična
	!=	Istina ako dva stringa nisu identična
	>	Istina ako je prvi string leksikogr. veći
	<	Istina ako je prvi string leksikogr. manji
	>=	Istina ako je prvi string lek. veći ili jed.
	<=	Istina ako je prvi string lek. manji ili jed
Ulaz/izlaz	>>	Za upotrebu sa ulaz. objektima i str. obj
	<<	Za upotrebu sa izlaz. objektima i str.obj
Pristup znakovima	[ ]	Pristup pojed. znak., bez r.c.; bolje at()
Spajanje	+	Spaja dva stringa; dodaje znak ili string

C++ `string` razred – neki funkcijski članovi

- traženje stringa u stringu, ubacivanje stringa u drugi string, brisanje dijelova stringa i druge operacije izvode se pomoću **funkcijskih članova klase `string`.**
- Poziv funkcijskog člana izvodi se upotrebom imena objekta, točke i imena funkcijskog člana. Ako je potrebno, drugi se objekt (string) predaje kroz listu argumenata.
- Obje u programu spomenute deklaracije pozivaju konstruktor za string klasu koji pridružuje navedeni string deklariranom objektu

Inicijalizacija C++ `string` objekta u deklaraciji – primjer 2.

```
#include <iostream>
#include <string>

using namespace std;

int main() {        //inicijalizacija string objekta u deklaraciji
   string s1 ("Nisu stigli na predstavu."), s2="na";
   int i;
   i=s1.find (s2);  //vraća (prvu) poziciju stringa s2 u s1 (i=12)
   s1.replace(i, 2, "za");
                    //mijenja 2 znaka od s1 u "pogledati" od pozicije i
   cout<<s1<<endl;
   s1.erase(15, 9); //briše 9 znakova stringa s1 od lokacije 15
   cout<<s1<<endl;
   s1.insert (15, "utakmicu");
                    //ubacuje "utakmicu" na 15. poziciju u stringu s1
   cout<<s1<<endl;
   return 0;
}
```

C++ `string` razred – funkcije `find` i `replace`

- Funkcija `find` je preopterećena. Njezina druga verzija ima dva argumenta. Drugi argument pokazuje indeks (cjelobrojnu vrijednost) kod koje će započeti pretraživanje.
- Ako string nije pronađen vraća se vrijednost `npos` koja je podatkovni član `string` razreda postavljen u deklaraciji na -1.
- Funkcija `replace` je preopterećena i njezina druga verzija ima dva argumenta više koji dozvoljavaju da dio stringa bude tekst koji mijenja postojeći.
- replace funkcija vraća referencu na objekt koji je poziva.
- `string` klasa automatski proširuje memorijsko područje za nove (ubačene) znakove što je prednost u odnosu na C-stringove.

Ostale C++ `string` funkcije – pretraživanje unutar stringa

Ime i argumenti	Povratna vrijednost	Opis
find (s,i)	Indeks prvog znaka od s u stringu, npos ako s nije u stringu.	Traži prvu pojavu od s počevši od i prema kraju stringa. Podrazumijevani i je 0.
rfind (s,i)	Indeks prvog znaka od s u stringu, npos ako s nije u stringu.	Traži prvu pojavu od s počevši od i prema početku stringa. Podrazumijevani i je 0.
find_first_of (s,i)	Indeks prvog znaka u oba stringa, npos ako u stringu nema znakova iz s.	Traži bilo koji znak iz s koji je u stringu počevši od i.
find_first_not_of (s,i)	Indeks prvog znaka stringa koji nije u s, npos ako su svi znakovi stringa u s.	Traži bilo koji znak u stringu koji nije u s počevši od i.
find_last_of (s,i)	Indeks posljednjeg znaka u oba stringa, npos ako niti jedan od znakova iz s nije u stringu.	Traži bilo koji znak koji je u oba stringa. Pretražuje unatrag počevši od i.
find_last_not_of (s,i)	Indeks posljednjeg znaka u stringu koji nije u s, npos ako su svi znakovi stringa u s.	Traži bilo koji znak u stringu koji nije u s unatrag počevši od i.
substr (i,n)	String od n znakova počevši sa i.	Vraća string objekt koji je podstring stringa (n znakova počevši od i).

Ostale C++ `string` funkcije – modificiranje stringa

Ime i argumenti	Povratna vrijednost	Opis
append (s,i,n)	Pozivajući objekt	Dodaje string od n znakova (od s) počevši sa i na kraj stringa.
assign (s,i,n)	Pozivajući objekt	Pridružuje n znakova počevši sa i od s u string.
erase (i,n)	Pozivajući objekt	Odstranjuje n znakova iz stringa počevši od i.
insert (i,s)	Pozivajući objekt	Ubacuje s u string počevši od i.
push_back (ch)	Nema	Dodaje znak stringu.
replace (i,n,s)	Pozivajući objekt	Mijenja n znakova u stringu počevši od i sa s.
resize (n,ch)	Pozivajući objekt	Mijenja veličinu stringa na n znakova (dulji ili kraći). Ako je dulji, ostatak se popunjava sa ch, ako je kraći, neki se znakovi gube.
swap (s)	Nema	Mijenja sadržaj stringa sa s.

Ostale C++ `string` funkcije – uspoređivanje stringova, pristup pojedinim znakovima, konverzija u druge tipove

Ime i argumenti	Povratna vrijednost	Opis
compare (i,n,s)	0 ako se pojavi potpuna podudarnost, >0 ako su karakteristike stringa leksikografski veće od s, <0 ako su karakteristike stringa leksikografski manje od s.	Uspoređuje n znakova počevši sa i u stringu sa s.
at (i)	Znak u stringu na poziciji i.	Koristi se za pristup pojedinom znaku u stringu. Slično [ ] u C stringovima. Javlja grešku ako dođe do pristupa iza kraja stringa, pa je zato sigurniji pristup znakovima od zagrada.
c_str ( )	Adresa početka C stringa - ekvivalenta pozivajućeg objekta. Nul znak automatski se dodaje stringu.	Kreira C string iz string objekta. Povratna vrijednost je konstanta, tako da string na kojeg pokazuje ne može biti promijenjen.
copy (polje,n,i)	Broj znakova kopiranih u polje.	Kopira n znakova počevši sa i iz stringa u polje znakova. Ne dodaje se nul znak, tako da polje nije važeći C string.
data ( )	Adresa početka ekvivalentnog znakovnog polja pozivajućeg objekta. Nema dodavanja nul znaka.	Kreira obično znakovno polje iz string objekta. Povratna vrijednost je konstanta i ne može biti promijenjena.

Ostale C++ `string` funkcije – značajke stringova

Ime i argumenti	Povratna vrijednost	Opis
capacity ( )	Kapacitet stringa.	Vraća kapacitet stringa ne zahtjevajući realokaciju.
empty ( )	0 ako string nema znakova.	Određuje da li je string prazan ili nije.
length ( )	Broj znakova u stringu.	Određuje stvarni broj znakova koje objekt čuva.
max_size ( )	Maksimalni mogući broj znakova za string objekt.	Određuje maksimalno moguću veličinu za string objekt.
reserve (n)	Nema.	Rezervira n znakova za string, ali ne reducira veličinu ispod tekućeg broja znakova.
size ( )	Broj znakova u stringu.	Slično kao length, vraća stvarni broj znakova spremljenih u objektu.

Vježbe 1

- Napraviti program koji učitava riječ pisanu malim tiskanim slovima i ispisuje
  - Koliko znakova je u učitanoj riječi
  - Učitanu riječ obrnuto (naopako)
  - Provjera je li učitana riječ palindrom (čita se jednako sa obje strane)
  - Koliko u toj riječi ima samoglasnika
  - Učitanu riječ velikim tiskanim slovima
  - Koliko se puta u učitanoj riječi pojavljuje neka zadana dvoslovna kombinacija, npr. “na”
  - Riječ koja se dobije tako da se u sredinu učitane riječi (ako ima paran broj slova) ubacuje neka druga zadana riječ, odnosno obriše se srednje slovo (ako riječ ima neparan broj slova)

Vježbe 2

- Napraviti program koji učitava riječ pisanu malim tiskanim slovima i ispisuje je tako da dva po dva znaka mijenja. Npr. Za riječ kamikaza ispis treba biti akimakaz
- Napraviti program koji za učitanu riječ i slovo ispisuje riječi koje se dobiju stavljanjem učitanog slova na različita mjesta u riječi, npr. Ako je riječ petak a slovo r ispis mora biti rpetak, pretak, pertak, petrak, petark i petakr
- Učitanu rečenicu ispisati tako da se okrene poredak riječi. Npr. Ovo je dosta lagano. Ispis treba biti: lagano dosta je ovo.
- Učitanu riječ ispisati tako da se znakovi ispišu sortirano. Npr. Za riječ PREMETALJKA, ispis treba biti AAEEJKLMPRT.
- Unzipati učitani tekst na slijedeći način. Ako je ulaz A3b2c4, izlaz mora biti AAAbbcccc.. Brojevi su uvijek jednoznamenkasti.

ZADACI

- Zbrajanje jednadžbi
- Ispravak krivo napisane riječi
- Ispis riječi rečenice prema njihovoj dužini
- Izračunavanje jednostavnog aritmetičkog izraza

C++ `string` razred – čitanje, ulaz s tipkovnice i iz datoteke

- Inicijalizacija stringa čitanjem s tipkovnice ili iz datoteke.
- Kao i za c-stringove koristi se `cin`, `getline` i `ignore` (ali različitog oblika).
- Kod deklariranja polja `string` objekata spremamo niz stringova (ne znakova).

Ulaz s tipkovnice i iz datoteke - 1

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


int main() {
   string s1, s2, s3[3];      //deklariranje dva string objekta i jednog
                              polja objekata
   ifstream uldat("C:\\TEKST.DAT");     //otvaranje ulazne datoteke
   int i;
   cout << "Unesite rijec " << endl;
   cin >> s1;                 //čitanje riječi (bez praznina ili "Enter") s
                              tipkovnice
                              //znakovi nakon praznine ne čitaju se; nema
                              setw!
   cin.ignore (1000,'\n');    //funkcija ignore uništava sve znakove koji bi
                               // mogli biti unijeti nakon praznine.
   cout << s1 << endl << endl;
```

Ulaz s tipkovnice i iz datoteke - 2

```
cout << "Unesite nekoliko redova. Završite sa #." << endl;
getline (cin, s2,  '#');   // Funkcija getline ima kao argumente cin,
                           //string objekt i terminator. Taj se poziv
                           //koristi za čitanje čitavih linija (više
                           //rečenica) s ulaza.
                           //Ulaz s tipkovnice pridružuje se objektu s2.
                           //Funkcija provjerava svaki znak koji unosi
                           //korisnik. Ako je unesen #, funkcija prestaje
                           // čitati ulaz.
cin.ignore (1000, '\n');   //Funkcija ignore koristi se za procesiranje
                           // "Enter" i svih drugih znakova koje bi
                           // korisnik unio nakon #.
cout << s2 << endl << endl;
for (i=0;i<3;i++){
      getline (uldat, s3[i]);       //možemo koristiti ifstream objekt
                                    //(uldat) kao argument za getline za
                                    // čitanje podataka iz datoteke.
                                    // Drugi argument je
      cout << s3[i];}               //jedan element polja string
                                    //objekata.
                                    //Korištenjem for petlje, možemo
                                    // spremiti jednu liniju datoteke po
                                    //elementu polja
return 0;}
```

Opis

- Količina memorije koja će se rezervirati za s1 automatski se prilagođava čuvanju svih znakova koje korisnik unese. `cin` je dobar za čitanje riječi, ali ne i čitave rečenice.
- `getline` čita znakove i sprema ih u objekt s2 (praznine i "Enter" uključeni su u string). Zato možemo čitati čitave odlomke ili stranice teksta. U pozivu se navodi i terminator (znak koji korisnik mora unijeti da bi označio kraj ulaza; čitamo ga, ali ne unosimo u objekt). Ako se čita **samo jedna linija ulaza**, terminator može biti i \n (inače podrazumijevana vrijednost posljednjeg argumenta funkcije `getline`).

Čitanje iz datoteke

- Sa `getline` može se pročitati čitav sadržaj datoteke.
- Uobičajena metoda koristi jednodimenzionalno polje `string` objekata.
- Svaka linija datoteke može biti čitana u drugi element polja upotrebom `for` petlje.
- U petlji se čitaju sve linije datoteke pridružene objektu `uldat` (jedna po jedna, treći argument u pozivu `getline` je podrazumijevani \n).
- `string` objekt veličinom se automatski prilagođava veličini linije datoteke koju čita tako da se programer ne mora brinuti o rezerviranju dovoljne količine memorije za svaku liniju.

C++ `string` razred – stringovi i funkcije

- Predaja `string` objekata i informacija o `string` objektima funkcijama
- Mnogo različitih oblika deklaracije može se koristiti sa `string` objektima.

Primjer programa

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

string f1 (string, const string[], string&, string[]);

int main() {        //deklaracija string objekata i polja objekata
   string s1="String objekt.", s2[3]={"Polje od ", "tri string ",
   "objekata."}, s3;
   string s4="Jedan string", s5[2]={"Izmijenjeno polje", "Izmijenjeno"};
   s3=f1(s1, s2, s4, s5);
   cout <<s3<<endl<<s4<<endl<<s5[0]<<endl<<s5[1]<<endl;
   return 0;
}

string f1 (string ss1, const string ss2[], string& ss4, string ss5[]); {
   ss4+=" izmijenjen.";
   ss5[0]+=", element nula.";
   ss5[1]+=" polje, element jedan.";
   return (ss1+ss2[0]+ss2[1]+ss2[2]);}
```

Opis programa

- Deklaracija funkcije pokazuje `string` objekte koji se na različite načine predaju funkciji.
- Funkcija vraća `string` objekt koji se pridružuje s3.
- Poziv funkcije sa `string` objektima različitih oblika.
- Tipovi u zaglavlju podudaraju se sa tipovima u pozivu.
- U funkciji su naredbe koje modificiraju `string` objekte s4 i s5[]. To je dozvoljeno jer je ss4 referenca za s4 i ss5 adresa od s5 (bez `const` kvalifikatora).

Opis deklaracije i poziva funkcije

- U listi argumenata funkcije f1:
  - `string`: označava da će kopija `string` objekta biti prenijeta funkciji f1.
  - `const string[]`: označava da će adresa polja `string` objekata biti prenijeta funkciji f1. `const` označava da f1 ne može modificirati sadržaj bilo kojeg `string` objekta u polju.
  - `string&`: označava da će u f1 biti kreirano drugo ime (alias) za `string` objekt. To znači da f1 može izravno modificirati `string` objekt.
  - `string[]`: označava da će f1 biti prenijeta adresa polja `string` objekata. Kako se ne koristi `const`, f1 može modificirati sadržaj objekata u polju.
- Tip povratne vrijednosti za funkciju je `string`, što znači da vraća `string` objekt.

Prijenos `string` objekata u listi argumenata

- Općenito, `string` objekti prenose se kao drugi objekti. Kad se tip `string` koristi u listi argumenata, prenosi se kopija.
- Ključna riječ `const` spriječava da elementi polja budu modificirani.
- Znak `&` označava referencu.
- Ako polje nije kvalificirano sa `const`, polje se može mijenjati.
- Važno je paziti na podudarnost tipova podataka u pozivu i zaglavlju funkcije.

Tijelo funkcije

- Unutar tijela funkcije možemo koristiti svaki argument na način koji odgovara njegovom tipu.
- String s4 i polje stringova s5[] funkcija će modificirati jer zaglavlje pokazuje da je ss4 referenca, a ss5[] prima adresu bez `const` kvalifikatora.
- Naredba `return` vraća `string` objekt sastavljen od četiri spojena `stringa` i taj se string u funkciji `main()` pridružuje stringu s3.

Vježbe

- Napravite program koji će demonstrirati upotrebu nekih funkcijskih članova klase string:

find (s,i)

rfind (s,i)

find_first_of (s,i)

find_first_not_of (s,i)

find_last_of (s,i)

find_last_not_of (s,i)

substr (i,n)

append (s,i,n)

erase (i,n)

insert (i,s)

push_back (ch)

replace (i,n,s)

resize (n,ch)

swap (s)

length ( )

****

Stringovi, klase i objekti - Upotreba funkcijskih članova za manipulaciju `string` objektima

- Upotreba `string` objekata i C-stringova kao podatkovnih članova klase.
- Podatkovni članovi su `private` kako bi bila osigurana enkapsulacija.
- To znači da je jedan način rada sa podacima sa accessor (pristupnim) (ili “get”) funkcijama.
- Koristimo funkcijski član (read_data) za inicijalizaciju svih podatkovnih članova.
- Podatkovne članove ispisujemo upotrebom cout i “get” funkcija.

Primjer programa

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

class Klasa1{
   private:
          char aa[20];
          string s1;
   public:
          char* get_aa();
          string get_s1();
          void read_data();};

char* Klasa1::get_aa () {
   return aa;}
//vraćanje adrese polja znakova aa[]
//tip povratne vrijednosti je char* što
//označava adresu prvog elementa polja
   aa[]

string Klasa1::get_s1 () {
   return s1;}
//vraćanje string objekta s1

void Klasa1::read_data () {
   cout << "Ime i prezime:"<<endl;
   getline (cin, s1);
   cout << "Telefonski broj:"<<endl;
   cin.getline (aa);}
//čitanje podataka

int main() {
   Klasa1 obj1;

   obj1.read_data;
   cout << obj1.get_s1 << endl
   <<obj1.get_aa()<<endl;
   return 0;}
```

Opis programa, definicija klase

- C string polje znakova i string objekt su privatni podatkovni članovi Klase1.
- Funkcije za manipuliranje stringovima također moraju biti članovi klase.
- Funkcije get_aa() i get_s1() pristupaju stringovima “get” funkcija vraća privatni podatkovni član.
- Funkcija read_data čita s tipkovnice oba string podatkovna člana.
- Kako main nije funkcijski član, stringovima (aa[] i s1) ne možemo pristupiti izravno, već moramo pristupati upotrebom pristupnih (accessor) funkcija – fundamentalno za objektno orijentirano programiranje jer osigurava da klasa kontrolira kako će se podacima moći pristupati i kako će ih se moći modificirati

Funkcijski članovi

- Za čitanje podataka koristimo funkciju getline.
- Funkcija getline ne vraća vrijednost (ima tip void).
- Kad se getline koristi sa string objektom, cin je argument, a kad se koristi sa C stringom, cin je pozivajući objekt
- s1 i aa ne moraju biti deklarirani u read_data jer je read_data funkcijski član
- “get” funkcije koje pristupaju podatkovnim članovima jednostavno vraćaju vrijednost podatkovnog člana
- Važno je da tip povratne vrijednosti za te funkcije bude konzistentan sa tipom podatkovnog člana. Kako ne možemo vratiti polje znakova iz get_aa(), moramo vratiti adresu polja upotrebom tipa char*.
  - Sa const se osigurati da se adresa neće koristiti za modificiranje sadržaja privatnih podatkovnog člana aa[]