not safe		safe
~~**char * D = new char[x];**~~ **char D[10];**	**char D[10];**	**char * D = new char[size_d];**
strcpy(D, S)	strcpy_s(D, S)	strcpy_s(D, size_d, S)
strncpy(D, S, size_s)	strncpy_s(D, S, size_s)	strncpy_s(D, size_d, S, size_s)

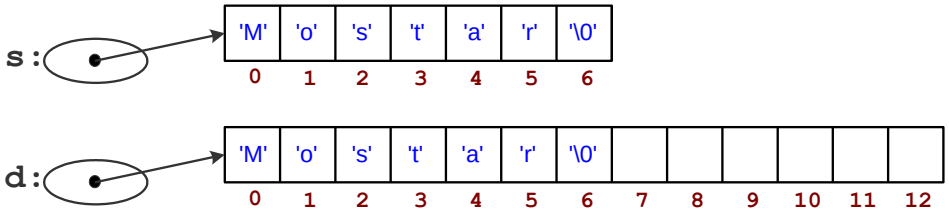
Zadatak 1.

Šta je rezultat slijedeća tri primjera.

*Primjer **a**:* niz *destination* je veći od niza *source*

s je dinamički niz d je statički niz	d je dinamički niz s je statički niz

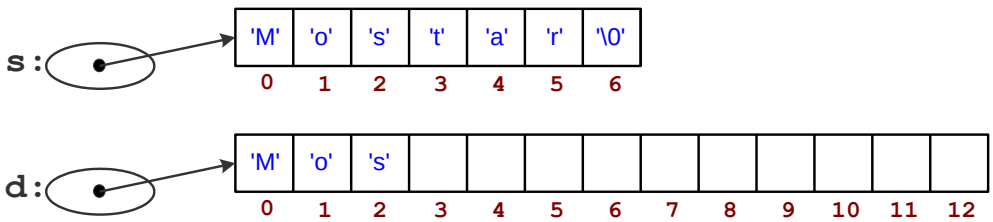
`char* s = new char[7];`
`cin >> s;`
`char d[13];`
 	
`strcpy(d, s);`
 	
`cout << strlen(s) << endl;`
`cout << strlen(d) << endl;`
	`char s[7];`
	`cin >> s;`
	`char* d = new char[13];`
	
	`strcpy(d, s);`
	
	`cout << strlen(s) << endl;`
	`cout << strlen(d) << endl;`



*Primjer **b**:* korištenje funkcije `strncpy`

s je dinamički niz d je statički niz	d je dinamički niz s je statički niz

`char* s = new char[7];`
`cin >> s;`
`char d[13];`
 	
`strncpy(d, s, 3);`
 	
`cout << strlen(s) << endl;`
`cout << strlen(d) << endl;`
	`char s[7];`
	`cin >> s;`
	`char* d = new char[13];`
	
	`strncpy(d, s, 3);`
	
	`cout << strlen(s) << endl;`
	`cout << strlen(d) << endl;`

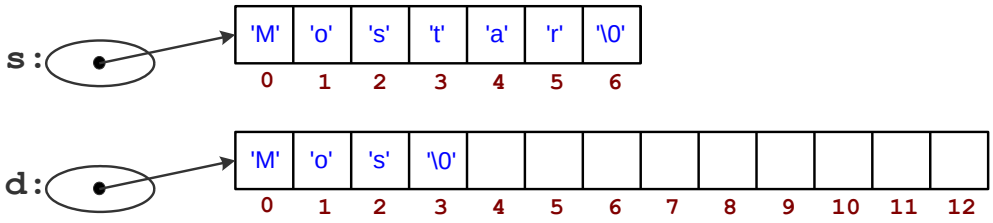


Ovdje je potrebno dodati karakter '\0' na kraj stringa. Slijedi ispravak.

bez rješenja

```
char* s = new char[7];
cin >> s;
char d[13];

strncpy(d, s, 3);
d[3] = '\0';

cout << strlen(s) << endl;
cout << strlen(d) << endl;
```

```
char s[7];
cin >> s;
char* d = new char[13];

strncpy(d, s, 3);
d[3] = '\0';

cout << strlen(s) << endl;
cout << strlen(d) << endl;
```

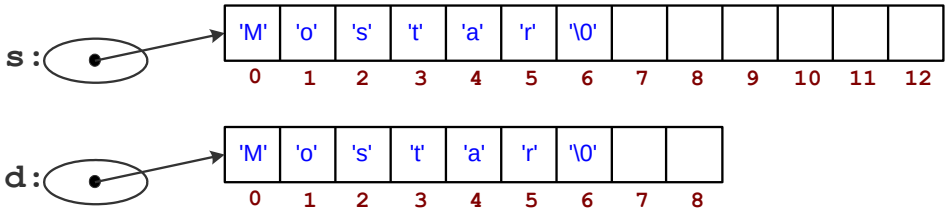


*Primjer **c**:* niz *destination* je manji od niza *source*, ali nije manji od *stringa* iz *source*

Da li će doći do greške?

```
char* s = new char[13];
cin >> s;
char d[9];

strcpy(d, s);

cout << strlen(s) << endl;
cout << strlen(d) << endl;
```

```
char s[13];
cin >> s;
char* d = new char[9];

strcpy(d, s,);

cout << strlen(s) << endl;
cout << strlen(d) << endl;
```



Zadatak 2

Pitanje: Šta će ispisati sljedeći kôd?

```
char p [20];

cout << strlen(p) << endl;
```

Zadatak 3

- Definišite pokazivač A na niz karaktera u dinamičkoj memoriji u kojeg ćete smjestiti ime i prezime.
- Definišite statički niz B dužine 20 u koji ćete unjeti mjesto rođenja
- A i B kopirajte u novi dinamički niz karaktera C.
- Reciklirajte oba dinamička niza.

Zadatak 4

Analizirajte sljedeći program.

```
1:   void main()
2:   {
3:          char* str1 = "ovo je prvi string u statickoj memoriji";
4:          char str2[] = "ovo je drugi string u statickoj memoriji";
5:
6:          char* p1;
7:          char* p2;
8:          p1 = str1;  //p1 pokazuje na string str1
9:          p2 = str2;  //p2 pokazuje na string str2
10:
11:         cout << str1 << endl;
12:         cout << str2 << endl;
13:         cout << p1 << endl;
14:         cout << p2 << endl;
15:  }
16:
```

Pitanje: Da li će se u linijama 11, 12, 13 i 14 ispisati adresa prvog elementa niza? Obrazložite.

Zadatak 5

Dovršite naredni program:

1) ispisati dati string od trećeg karaktera
2) ispisati prvi karakter stringa
3) ispisati zadnji karakter stringa
4) ispisati zadnjih 10 karaktera stringa

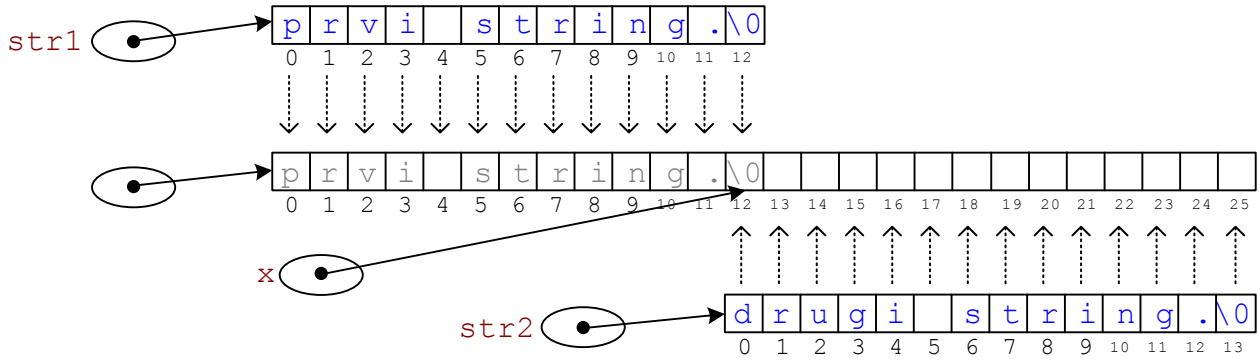
```
1:   void main()
2:   {
3:          char* str1 = "ovo je prvi string u statickoj memoriji";
             ...
```

Rješenje	

Zadatak 6

Deklarišite dva stringa (str1 i str2). Zatim ih iskopirajte (spojite) u treći string str3 i ispišite ga na ekran.

a) koristite funkciju `strcat`
b) koristite funkciju strcpy (bez korištenja funkcije `strcat`)



Rješenje - a

Rješenje - b

Zadatak 8

a) Implementirajte funkciju `dodjeli_str` koja će alocirati niz u dinamičkoj memoriji i kopirati ulazni string u taj niz, npr.

```
1:   void main()
2:   {
3:         char* str1 = dodjeli_str("Ovo je neki string");
4:         //funkcija vrši alokaciju i kopiranje
5:
6:         cout << str1 << endl;
7:
8:         delete [] str1; //dealokacija
9:   }
```

Rješenje:

b) Primjenite funkciju `dodjeli` u sljedećem programu.

```
1:   void main()
2:   {
3:          char* str1;
4:
5:          str1 = new char[strlen("Ovo je neki string") + 1];
6:          strcpy(str1, "Ovo je neki string");
7:          cout << str1 << endl;
8:
9:          char* str2 = new char[strlen(str1) + 1];
10:         strcpy(str2, str1);
11:         cout << str2 << endl;
12:
13:         delete [] str1;
14:         delete [] str2;
15:  }
```

bez rješenja

Zadatak 9

Implementirajte funkciju `dodaj_str` koja će prvom stringu dodati drugi string.

```
1:   void main()
2:   {
3:          char* str1 = dodjeli_str("Prvi niz. ");
4:          dodaj_str(str1, "Ovo je drugi niz");
5:
6:          cout << str1 << endl; //ispis: "Prvi niz. Ovo je drugi niz."
7:          delete [] str1;
8:   }
```

Rješenje

Zadatak 10:

Definišite funkciju "`void dodjeli_str(char* &d, char* b)`" koja ima istu funkcionalnost kao "`char* dodjeli_str(char* s)`" s tim da izlaz bude pomoću reference a ne pomoću naredbe `return`.

Nemojte brisati prvu definiciju funkcije, jer je moguće imati više funkcija sa istim imenom a koje se razliku po broju ili tipu parametara.

Zadatak 11:

a) Definište funkciju `spoji_str` koja će kreirati izlazni string sastavljen od dva ulazna stringa. Izlazna vrijednost neka bude pomoću naredbe `return`.

Rješenje

b) Ponovo definište funkciju `spoji_str` koja će kreirati izlazni string sastavljen od tri ulazna stringa. Izlazna vrijednost neka bude pomoću naredbe `return`.

Rješenje

Kopirajte vaše četiri funkcije sa njihovim prototipovima u jedan fajl i taj fajl snimite u folder gdje se nalaze i drugi *header*-fajlovi ili negdje na disk tako da možete ubuduće koristiti vaše funkcije kao gotove funkcije, npr.:

bez rješenja

```
1:   #include <iostream>
2:   using namespace std;
3:   #include "c:\temp\str_funkcije.h"
4:
5:   void main()
6:   {
7:          char* p1 = dodjeli_str("_Fit for FIT");
8:
9:          char* p2;
10:         dodjeli_str(p2, "He says:");
11:         dodaj_str(p2, p1);
12:         cout << "p2 = " << p2 << endl;
13:
14:         char* p3 = spoji_str(p2, " > ", p1);
15:         //...nemojte zaboraviti dealocirati nizove p1, p2 i p3...
```

Zadatak 12 – Tekst Editor

Dovršite program

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

char crt[] = "\n-----------------------------------------------------\n";

void OslobodiMemoriju(char * &tekst)
{
   //dealocirati tekst
}
void Informacije(char * tekst)
{
       int razmaci = 0, brojevi = 0, velika = 0, mala = 0, interpunkcijski = 0;




       cout << crt << "\t\t::INFO::" << crt;
       cout << crt << "Tekst: " << tekst << crt;
       cout << "Niz ima " << strlen(tekst) << " karaktera." << crt;
       cout << "Razmaka: \t\t\t" << razmaci << endl;
       cout << "Brojeva: \t\t\t" << brojevi << endl;
       cout << "Velikih slova: \t\t\t" << velika << endl;
       cout << "Malih slova: \t\t\t" << mala << endl;
       cout << "Interpunkcijskih znakova: \t" << interpunkcijski;
       cout << crt << "Info: Informacije prikazane...." << crt;
}

void DodajTekst(char * &tekst)
{
       //stari tekst + razmak + novi tekst
}

void Pretraga(char * tekst)
{
  //...
}

void UnosTeksta(char *& tekst)
```

bez rješenja

```
{
   //...
}

int PrikaziMeni() {

	int izbor = 1;
	do {
		cout << crt << "\t\t::MENI::" << crt;
		cout << "1. Unos novog teksta" << endl;
		cout << "2. Dodavanje teksta" << endl;
		cout << "3. Informacije o tekstu" << endl;
		cout << "4. Pretraga" << endl;
		cout << "5. Zatvori editor" << endl;
		cout << "Unesite vas izbor: ";
		cin >> izbor;
		cin.ignore();
		system("cls");
	} while (izbor < 1 || izbor>5);
	return izbor;
}

void main() {

	int izbor = 0;
	char * tekst = NULL;
	do {
		cout << crt << "\t\t::TEKST EDITOR::";
		izbor = PrikaziMeni();
		switch (izbor) {
		case 1:
			UnosTeksta(tekst); break;
		case 2:
			DodajTekst(tekst); break;
		case 3:
			Informacije(tekst); break;
		case 4:
			Pretraga(tekst); break;
		}
		system("pause>0");
		system("cls");
	} while (izbor != 5);

	if (tekst != NULL)
		OslobodiMemoriju(tekst);

	cout << crt;
}
```


Zadatak 13 – Pametni pokazivači

Prepravite program tako da umjesto običnih pokazivača koristite pametni pokazivač `shared_ptr`

Primjer definisana pokazivača

bez rješenja

```
shared_ptr<Datum> date (new Datum);
```

Primjer pristupa članovima objekta na koji pokazuje pokazivač

date->d

jednostavan primjer

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

struct Datum
{
	int d;
	int m;
	int g;
};

void info1(Datum* date){
	if (date == nullptr)
		return;
	cout << (*date).d << "." << date->m << "." << date->g;
}
void info2(shared_ptr<Datum> date){
	if (date == nullptr)
		return;
	cout << (*date).d << "." << date->m << "." << date->g;
}
void main() {
	Datum* d1 = new Datum;
	d1->d = 30;   d1->m = 3; d1->g = 2017;
	info1(d1);
	delete d1;
	shared_ptr<Datum> d2(new Datum);
	d2->d = 30;   d2->m = 3;d2->g = 2017;
	info2(d2);
}
```

Code za zadatak

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

//----------struktura DATUM-------------
struct Datum
{
	int d;
	int m;
	int g;
};

void info(Datum* date)
{
	if (date == nullptr)
		return;

	cout << (*date).d << "." << date->m << "." << date->g;
}
Datum* ucitaj_datum()
```

bez rješenja

```
{
	Datum* date = new Datum;
	cout << "Unesite dan, mjesec, godinu:" << endl;
	cin >> date->d >> date->m >> date->g;
	return date;
}


//----------struktura OSOBA-------------
struct Osoba
{
	char ime[10];
	int id;
};

void info(Osoba* o)
{
	if (o == nullptr)
		return;
	cout << "Id: \t" << (*o).id << ", Ime: " << o->ime;
}

Osoba* odaberite_osobu(Osoba* osobe, int max)
{
	for (int i = 0; i<max; i++)
	{
		info(osobe + i);
		//ili
		//info(&osobe[i]);
		//ili
		//info(&(*(osobe+i)));

		cout << endl;
	}

	do
	{
		cout << "Unesite ID osobe: ";
		int x;
		cin >> x;

		for (int i = 0; i < max; i++)
		{
			if (osobe[i].id == x)
				return osobe + i;
				//ili
				// return &osobe[i];
		}
		cout << "Neispravan Id" << endl;
	} while (true);
}

const int max_s = 10;
Osoba studenti[max_s] = {
	{ "Student J", 101 },
	{ "Student I", 102 },
	{ "Student H", 103 },
	{ "Student G", 104 },
	{ "Student F", 105 },
	{ "Student E", 106 },
	{ "Student D", 107 },
	{ "Student C", 108 },
	{ "Student B", 109 },
	{ "Student A", 110 },
};
```

bez rješenja

```
const int max_i = 5;
Osoba ispitivaci[max_i] = {
	{ "Nina B.", 534 },
	{ "Jasmin A.", 435 },
	{ "Denis M.", 256 },
	{ "Emina J.", 325 },
	{ "Zanin V.", 914 },
};

struct PrijavaZaIspit
{
	Osoba* student;
	Datum* datum_prijave;
	Datum* datum_odjave;
	int ocjena;
};

const int max_p = 100;
struct Ispit
{
	Osoba* ispitivac;
	Datum* datum_ispita;
	int brojac_prijavljenih;
	char predmet[10];
	PrijavaZaIspit* prijave = new PrijavaZaIspit[max_p]; //umjesto PrijavaZaIspit
prijave[max_p];
};


Ispit* dodaj_ispit()
{
	Ispit* i = new Ispit;
	(*i).brojac_prijavljenih = 0;
	//ili
	//i->brojac_prijavljenih = 0;

	cout << "Unesite naziv predmeta (bez razmaka)" << endl;
	cin >> i->predmet;

	cout << "Unesite datum ispita: ";
	i->datum_ispita = ucitaj_datum();

	cout << "Ispitivac: ";
	i->ispitivac = odaberite_osobu(ispitivaci, max_i);
	return i;
}


void info(Ispit* i)
{
	if (i == nullptr)
		return;

	cout << "Predmet: " << i->predmet;

	cout << ", \tDatum ispita: ";
	info(i->datum_ispita);

	cout << ", \tIspitivac: ";
	info(i->ispitivac);

	cout << ", \tBroj prijavljenih: " << i->brojac_prijavljenih << endl;
}
```

bez rješenja

```
Ispit* odaberite_ispit(Ispit* ispiti, int max)
{
	for (int i = 0; i<max; i++)
	{
		cout << i + 1 << ": ";
		info(ispiti + i);
		//ili
		//info(&ispiti[i]);
	}
	cout << "Unesite RB ispita: ";
	int x;
	cin >> x;
	return &ispiti[x - 1];
	//ili
	//return ispiti + x - 1;
}


void main()
{
	const int max_ispiti = 10;
	Ispit* ispiti = new Ispit[max_ispiti];
	int brojac_ispita = 0;
	int x;
	do
	{
		cout << "1. Ispiti - prikazi sve" << endl;
		cout << "2. Ispiti - dodaj novi (zakazi novi ispit)" << endl;
		cout << "3. Prijavljeni za ispit: prikazi sve (za ispit x)" << endl;
		cout << "4. Prijavljeni za ispit: dodaj novu prijavu (za ispit x)" << endl;
		cout << "5. Evidentiraj sve ocjene (za ispit x)" << endl;
		cout << "0. EXIT" << endl;
		cin >> x;
		if (x == 1)
		{
			for (int i = 0; i<brojac_ispita; i++)
			{
				info(ispiti + i);
				//ili
				//info(&ispiti[i]);
			}
		}
		if (x == 2)
		{
			Ispit* i = dodaj_ispit();
			ispiti[brojac_ispita] = *i;
			brojac_ispita++;
		}
		if (x == 3)
		{
			Ispit* i = odaberite_ispit(ispiti, brojac_ispita);
			for (int j = 0; j<i->brojac_prijavljenih; j++)
			{
				Osoba* s = i->prijave[j].student;
				Datum* dP = i->prijave[j].datum_prijave;
				Datum* dO = i->prijave[j].datum_odjave;
				info(s);
				cout << " Datum prijave: ";
				info(dP);
				cout << " Datum odjave: ";
				info(dO);
				cout << endl;
			}
		}
		if (x == 4)
		{
```

bez rješenja

```
                        Ispit* i = odaberite_ispit(ispiti, brojac_ispita);
                        PrijavaZaIspit* novaP = new PrijavaZaIspit;
                        novaP->student = odaberite_osobu(studenti, max_s);
                        novaP->datum_prijave = ucitaj_datum();
                        novaP->datum_odjave = nullptr;
                        i->prijave[i->brojac_prijavljenih++] = *novaP;
                }
                if (x == 5)
                {
                        Ispit* i = odaberite_ispit(ispiti, brojac_ispita);
                        for (int j = 0; j<i->brojac_prijavljenih; j++)
                        {
                                info(i->prijave[j].student);
                                cout << "Unesite ocjenu sa ispita: ";
                                cin >> i->prijave[j].ocjena;
                        }
                }
        } while (x != 0);

}
```