Uvod u programiranje

- predavanja -

siječanj 2021.

24. Datoteke

- 2. dio -

Datoteke

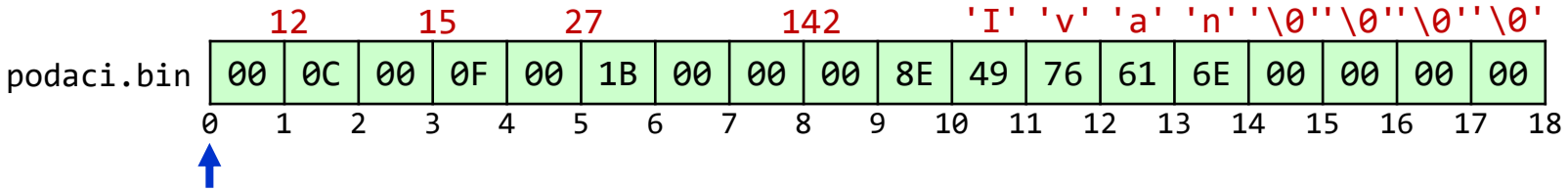
Indikator pozicije

Indikator pozicije u datoteci

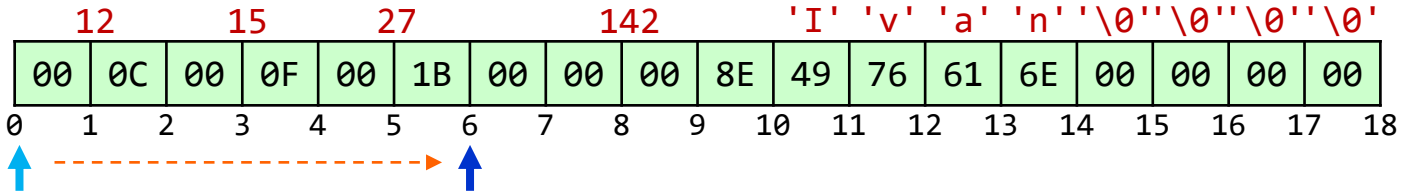
```
short polje[3];
struct osoba_s {
   int rbr;
   char ime[7 + 1];
};
struct osoba_s stud;
```

- Podaci se uvijek čitaju ili pišu počevši od mjesta u datoteci na koje trenutačno pokazuje indikator pozicije (*file position indicator*)

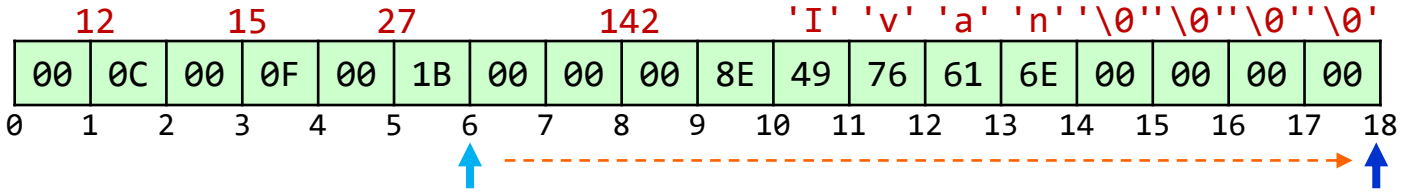
```
FILE *bin = fopen("podaci.bin", "rb");
```




```
fread(&polje[0], sizeof(polje), 1, bin);
```




```
fread(&stud, sizeof(stud), 1, bin);
```

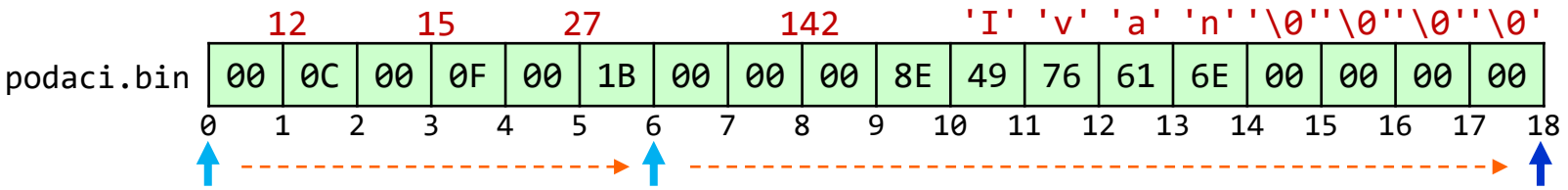


ftell <stdio.h>

```
long ftell(FILE *stream);
```

- rezultat funkcije: vrijednost indikatora pozicije za tok `stream`, tj. udaljenost od početka datoteke za koju je otvoren tok na kojeg pokazuje `stream`, izražena u broju bajtova
  - ako je indikator pozicije na samom početku datoteke, rezultat je `0L`
  - u slučaju pogreške, funkcija vraća `-1L`

```
FILE *bin = fopen("podaci.bin", "rb");
```




```
printf("%ld", ftell(bin));                    0
fread(&polje[0], sizeof(polje), 1, bin);
printf("%ld", ftell(bin));                    6
fread(&stud, sizeof(stud), 1, bin);
printf("%ld", ftell(bin));                    18
```

fseek <stdio.h>

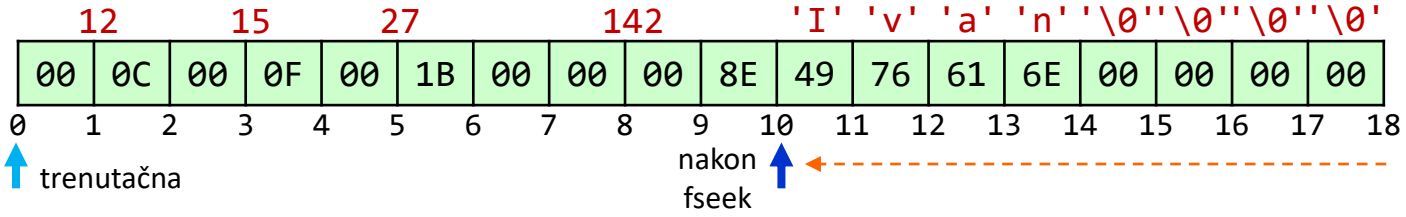
```
int fseek(FILE *stream, long offset, int whence);
SEEK_SET                makro:  u odnosu na početak datoteke
SEEK_CUR                        u odnosu na trenutačnu poziciju
SEEK_END                        u odnosu na kraj datoteke
```

- indikator pozicije za tok `stream` pomiče na poziciju koja je za `offset` bajtova udaljena od:
  - početka datoteke (za `whence` = `SEEK_SET`)
  - tekuće pozicije (za `whence` = `SEEK_CUR`)
  - kraja datoteke (za `whence` = `SEEK_END`)
- rezultat funkcije:
  - u slučaju pogreške vraća cijeli broj različit od nule, inače nulu

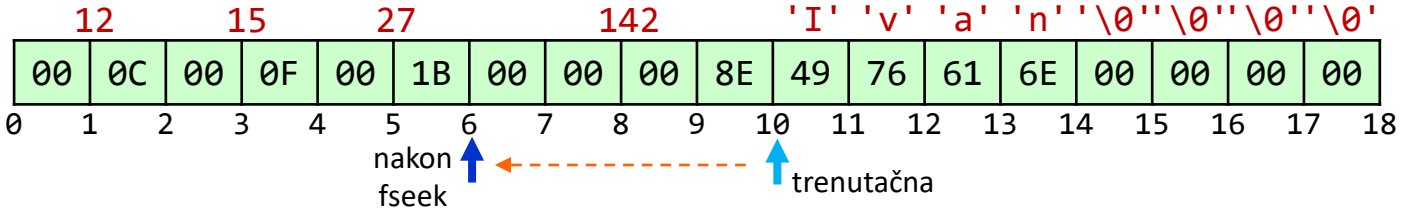
Primjer

```
short polje[3];
struct osoba_s {
   int rbr;
   char ime[7 + 1];
};
struct osoba_s stud;
```

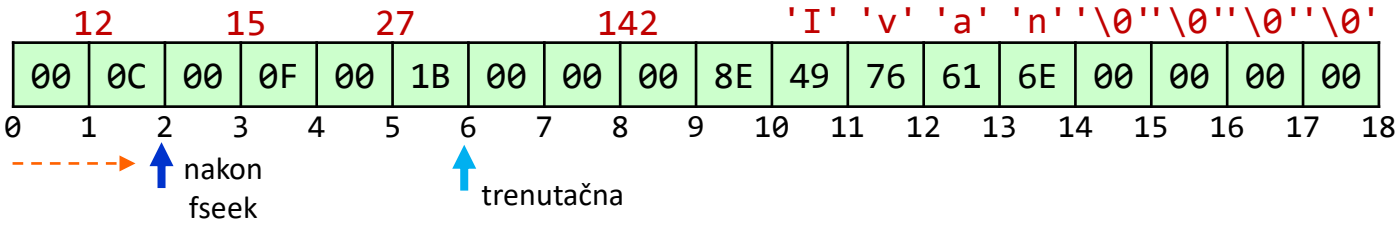
```
fseek(bin, -8L, SEEK_END);
```




```
fseek(bin, -1L * sizeof(stud.rbr), SEEK_CUR);
```




```
fseek(bin, (long)sizeof(polje[0]), SEEK_SET);
```



Datoteke

Slijedni i direktni pristup zapisima u datoteci

Slijedni pristup zapisima u datoteci

- zapisima se pristupa redom, slijedno
  - jedan po jedan zapis čita se od početka prema kraju datoteke, dok se ne pročitaju svi zapisi ili dok se ne pročita jedan ili više zapisa koje je potrebno obraditi
    - kolokvijalno: slijedna obrada datoteke
  - slijedni pristup podacima je primjenjiv u svim datotekama (neovisno imaju li fiksne duljina zapisa i na koji način su zapisi pozicionirani u datoteci), ali nije uvijek dovoljno efikasan
    - slijedni pristup se koristi kada se trebaju obraditi svi ili većina zapisa u datoteci ili kada direktni pristup do zapisa nije moguć

```
Ivan 10.35
Ana 0.5
Ante 2.1
```

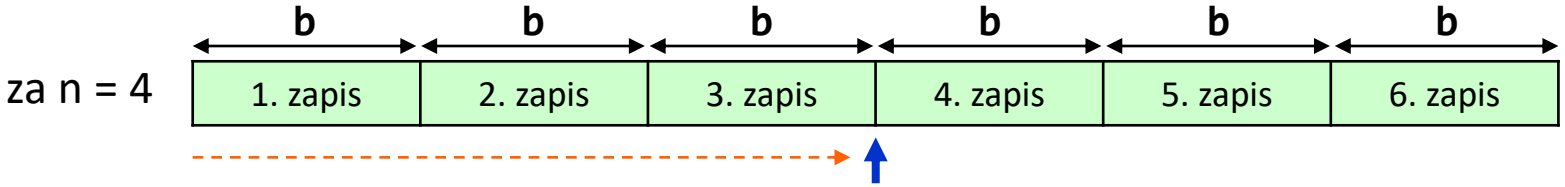
```
fscanf(tok, "%s %f", ime, &bodovi);   čitanje 1. zapisa
fscanf(tok, "%s %f", ime, &bodovi);   čitanje 2. zapisa
fscanf(tok, "%s %f", ime, &bodovi);   čitanje 3. zapisa
```

    - nije moguće pročitati treći zapis bez da se pročitaju prethodni

Direktni pristup zapisima u datoteci

- zapisima se pristupa izravno, direktno, na temelju rednog broja zapisa (kolokvijalno: direktna obrada datoteke)
  - zapisi u datoteci **moraju** biti fiksne duljine (npr. `b` bajtova)
  - iz rednog broja zapisa, npr. `n` (prvi zapis u datoteci ima redni broj 1), izračuna se pozicija na kojoj se u datoteci nalazi dotični zapis
  - indikator pozicije se pomakne na početak tog zapisa

```
fseek(tok, (long)(n - 1) * b, SEEK_SET);
```



  - zapis se zatim pročita (ili se zapiše nova vrijednost zapisa)
- Primjer: pomak indikatora pozicije na početak n-tog zapisa datoteke u kojoj zapisi odgovaraju strukturi `zapis_s`

```
fseek(tok, (long)(n - 1) * sizeof(struct zapis_s), SEEK_SET);
```

Direktni pristup zapisima u datoteci

- mogućnost direktnog pristupa n-tom zapisu u datoteci praktično je iskoristiva samo onda kada redni broj zapisa odgovara nekom ključu potrage ili se iz ključa potrage može izračunati, npr:
  - zapis o osobi s matičnim brojem n uvijek je smješten kao n-ti zapis u datoteci (ključ potrage u ovom slučaju je matični broj osobe)
- moguće je da će neki zapisi biti "prazni", npr.
  - ako su u datoteku upisani podaci o 100 osoba, a ne postoje osobe s matičnim brojevima 2, 17, 33, 34

- Primjer (kada se redni broj zapisa može *izračunati* iz ključa potrage):
  - ako datoteka sadrži zapise o mjestima (poštanski broj i naziv mjesta, a raspon poštanskih brojeva je od 10 000 do 60 000), kako omogućiti direktni pristup do zapisa na temelju zadanog poštanskog broja?

Primjer

- Programski zadatak
  - svaki zapis tekstne datoteke `drzave.txt` sadrži numeričku šifru, kraticu naziva i naziv države (međusobno su odvojeni znakom praznine). Šifra države je pozitivni cijeli broj, a naziv države nije dulji od 40 znakova
  - napisati program kojim će se na zaslon ispisati:
    - broj država koje su upisane u datoteku
    - najveća šifra države

`drzave.txt`

```
1 PR Puerto Rico↵
248 AX Aland Islands↵
3 AL Albania↵
24 AO Angola↵
...
897 TG Togo↵
898 PN Pitcairn↵
894 ZM Zambia↵
79 ZW Zimbabwe↵
895 CK Cook Islands↵
732 EH Western Sahara↵
```

- Analiza
  - potrebno je pročitati sve zapise datoteke
    - prikladno je koristiti slijedni pristup zapisima

Rješenje

```
...
int sifDrz, maxSif, brojac = 0;

FILE *tok = fopen("drzave.txt", "r");

while (fscanf(tok, "%d %*s %*[^\n]", &sifDrz) == 1) {
   ++brojac;
   if (brojac == 1 || sifDrz > maxSif) {
      maxSif = sifDrz;
   }
}
printf("Broj zapisa = %d, najveca sifra = %d", brojac, maxSif);

fclose(tok);
...
```

Primjer

- Programski zadatak
  - s tipkovnice učitati jedan cijeli broj. Ako u datoteci `drzave.txt` postoji država sa šifrom koja odgovara učitanom broju, na zaslon ispisati naziv te države. Inače, ispisati poruku "Nema drzave s tom sifrom"

`drzave.txt`

```
1 PR Puerto Rico↵
248 AX Aland Islands↵
3 AL Albania↵
24 AO Angola↵
...
897 TG Togo↵
898 PN Pitcairn↵
894 ZM Zambia↵
79 ZW Zimbabwe↵
895 CK Cook Islands↵
732 EH Western Sahara↵
```

- Analiza
  - jedini način kako pristupiti traženom zapisu jest redom čitati zapis po zapis datoteke i pročitanu šifru uspoređivati s traženom šifrom
  - ponavljati dok se ne pronađe zapis s odgovarajućom šifrom ili dok se ne pročitaju svi zapisi
  - postupak nije efikasan (slijedna obrada), ali u ovom slučaju jedini moguć

Rješenje

```
...
int trazimSifru, sifDrz;
char naz[40 + 1];

FILE *tok = fopen("drzave.txt", "r");

printf("Upisite sifru drzave > ");
scanf("%d", &trazimSifru);

int procitano;
do {
   procitano = fscanf(tok, "%d %*s %[^\n]", &sifDrz, naz);
} while (sifDrz != trazimSifru && procitano == 2);

if (sifDrz == trazimSifru && procitano == 2) {
   printf("%s\n", naz);
} else {
   printf("Nema drzave s tom sifrom\n");
}

fclose(tok);
...
```

Primjer

- Programski zadatak
  - Vrlo slično prethodnom zadatku: s tipkovnice učitavati po jedan cijeli broj dok god upisani cijeli broj odgovara šifri neke od država u datoteci `drzave.txt`. Ako postoji država sa šifrom koja odgovara učitanom broju, na zaslon ispisati naziv te države, inače, ispisati poruku "Nema drzave s tom sifrom" i prekinuti daljnje učitavanje cijelih brojeva.

`drzave.txt`

```
1 PR Puerto Rico↵
248 AX Aland Islands↵
3 AL Albania↵
24 AO Angola↵
...
897 TG Togo↵
898 PN Pitcairn↵
894 ZM Zambia↵
79 ZW Zimbabwe↵
895 CK Cook Islands↵
732 EH Western Sahara↵
```

Neispravno rješenje

```
...
FILE *tok = fopen("drzave.txt", "r");

while (1 == 1) {
   printf("Upisite sifru drzave > ");
   scanf("%d", &trazimSifru);

   int procitano;
   do {
      procitano = fscanf(tok, "%d %*s %[^\n]", &sifDrz, naz);
   } while (sifDrz != trazimSifru && procitano == 2);

   if (sifDrz == trazimSifru && procitano == 2) {
      printf("%s\n", naz);
   } else {
      printf("Nema drzave s tom sifrom\n");
      break;
   }
}
...
```

Nakon pretrage datoteke za jednu šifru učitanu s tipkovnice, indikator pozicije u toku **tok** ostao je iza pronađenog zapisa ili na kraju datoteke (ako zapis s odgovarajućom šifrom nije pronađen). To znači da pretraga za sljedeću šifru države učitanu s tipkovnice ne bi počela od prvog zapisa u datoteci.

Ispravno rješenje

```
...
FILE *tok = fopen("drzave.txt", "r");

while (1 == 1) {
   printf("Upisite sifru drzave > ");
   scanf("%d", &trazimSifru);

   int procitano;
   do {
      procitano = fscanf(tok, "%d %*s %[^\n]", &sifDrz, naz);
   } while (sifDrz != trazimSifru && procitano == 2);

   if (sifDrz == trazimSifru && procitano == 2) {
      printf("%s\n", naz);

      fseek(tok, 0L, SEEK_SET);     // indikator pozicije vrati na početak datoteke
   } else {
      printf("Nema drzave s tom sifrom\n");
      break;
   }
}
...
```

Nakon pretrage datoteke za jednu šifru učitanu s tipkovnice, indikator pozicije u toku **tok** vraća se na početak toka, kako bi pretraga za sljedeću šifru države učitanu s tipkovnice mogla početi od prvog zapisa u datoteci.

Primjer

drzave.txt

```
1 PR Puerto Rico↵
248 AX Aland Islands↵
3 AL Albania↵
...
```

- Programski zadatak
  - zapise iz tekstne datoteke `drzave.txt` treba prepisati u binarnu datoteku `drzave.bin`
  - svaki zapis datoteke `drzave.bin` treba sadržavati
    - šifru (int), dvoslovnu kraticu (2+1 znak) i naziv države (40+1 znak)
    - redni broj zapisa u datoteci `drzave.bin` mora odgovarati šifri države

0 →	1	PR	Puerto Rico
48 →	0000	000	0000000000000000
96 →	3	AL	Albania
144 →	...		
42912 →	895	CK	Cook Islands
42960 →	0000	000	0000000000000000
43008 →	897	TG	Togo
43056 →	898	PN	Pitcairn
43104 →			

- prazni zapisi (npr. ne postoji država sa šifrom 2 ili država sa šifrom 896) sadrže vrijednosti 0. Ako se iz datoteke pročita zapis kojem je šifra nula, to znači da takvog zapisa "nema".
- kako onda "obrisati" zapis? Vrijednost šifre postaviti na nulu
- ako bi se u datoteku na odgovarajuću poziciju upisao npr. zapis s rednim brojem 900, sustav bi prostor zapisa s rednim brojem 899 sam popunio nulama

Rješenje

```
...
struct drz_s {
   int sifDrz;
   char krat[2 + 1];
   char naz[40 + 1];
} drzava;

FILE *ulaz = fopen("drzave.txt", "r");
FILE *izlaz = fopen("drzave.bin", "wb");

while (fscanf(ulaz, "%d %s %[^\n]",
              &drzava.sifDrz, drzava.krat, drzava.naz) == 3) {
   fseek(izlaz, (long)(drzava.sifDrz - 1) * sizeof(drzava), SEEK_SET);
   fwrite(&drzava, sizeof(drzava), 1, izlaz);
}

fclose(ulaz);
fclose(izlaz);
...
```

Primjer

- Programski zadatak
  - svaki zapis datoteke `drzave.bin` sadrži
    - šifru (int), dvoslovnu kraticu (2+1 znak) i naziv države (40+1 znak)
    - redni broj zapisa u datoteci `drzave.bin` odgovara šifri države
  - s tipkovnice učitati jedan cijeli broj. Ako u datoteci `drzave.bin` postoji država sa šifrom koja odgovara učitanom broju, na zaslon ispisati naziv te države. Inače, ispisati poruku "Nema drzave s tom sifrom"

- Analiza
  - iz šifre države lako je izračunati poziciju (redni broj bajta) na kojoj započinje zapis o toj državi (ako takav zapis postoji). Koristiti direktni pristup zapisu
    - u ovom primjeru bilo bi **vrlo pogrešno** koristiti slijedni pristup zapisu
  - postaviti indikator pozicije na početak zapisa i pročitati ga. Ako je pročitana šifra jednaka nuli, tada zapisa s traženom šifrom u datoteci nema

Rješenje

U binarnim datotekama treba čitati (također i pisati) uvijek cijeli zapis, koristeći pri tome strukturu. Bilo bi pogrešno čitanje obaviti ovako:

```
fread(&drzava.sifDrz, sizeof(drzava.sifDrz), 1, tok);
fread(drzava.krat, sizeof(drzava.krat), 1, tok);
fread(drzava.naz, sizeof(drzava.naz), 1, tok);
```

```
...
struct drz_s {
   int sifDrz;
   char krat[2 + 1];
   char naz[40 + 1];
} drzava;
int trazimSifru;

scanf("%d", &trazimSifru);

FILE *tok = fopen("drzave.bin", "rb");
fseek(tok, (long)(trazimSifru - 1) * sizeof(drzava), SEEK_SET);
fread(&drzava, sizeof(drzava), 1, tok);

if (drzava.sifDrz == trazimSifru) {
   printf("%s", drzava.naz);
} else {
   printf("Nema drzave s tom sifrom");
}

fclose(tok);
...
```

Primjer

- Programski zadatak
  - svaki zapis datoteke `drzave.bin` sadrži
    - šifru (int), dvoslovnu kraticu (2+1 znak) i naziv države (40+1 znak)
    - redni broj zapisa u datoteci `drzave.bin` odgovara šifri države
  - s tipkovnice učitati niz od dva znaka. Ako u datoteci `drzave.bin` postoji država s kraticom koja odgovara učitanom nizu, na zaslon ispisati šifru i naziv te države. Inače, ispisati poruku "Nema drzave s tom kraticom"

- Analiza
  - ključ potrage ovdje je kratica države i iz nje nije moguće odrediti redni broj zapisa, pa posljedično niti poziciju zapisa u datoteci. Iako neefikasno, jedini način na koji se zadatak može riješiti jest koristiti slijedni pristup
    - redom čitati zapise i uspoređivati sa zadanom kraticom

Rješenje

```
...
struct drz_s {
   int sifDrz;
   char krat[2 + 1];
   char naz[40 + 1];
} drzava;
char trazimKrat[2 + 1];
scanf("%s", trazimKrat);

FILE *tok = fopen("drzave.bin", "rb");
while (fread(&drzava, sizeof(drzava), 1, tok) == 1) {
   if (drzava.sifDrz != 0 && strcmp(drzava.krat, trazimKrat) == 0) {
      break;
   }
}

if (strcmp(drzava.krat, trazimKrat) == 0) {
   printf("%d %s", drzava.sifDrz, drzava.naz);
} else {
   printf("Nema drzave s tom kraticom");
}

fclose(tok);
```

Neispravno rješenje

```
…
…

for (int i = 0; i < 100000; ++i) {
   fseek(tok, (long)(i - 1) * sizeof(drzava), SEEK_SET);
   fread(&drzava, sizeof(drzava), 1, tok);
   if (drzava.sifDrz != 0 && strcmp(drzava.krat, trazimKrat) == 0) {
      break;
   }
}
if (strcmp(drzava.krat, trazimKrat) == 0) {
   printf("%d %s", drzava.sifDrz, drzava.naz);
} else {
   printf("Nema drzave s tom kraticom");
}
fclose(tok);
```

Primjer

- Programski zadatak
  - u tekstnoj datoteci `kupljeno.txt` upisani su podaci o kupljenim artiklima. Zapis datoteke sadrži šifru artikla (4 znamenke) i broj kupljenih komada tog artikla (2 znamenke)

```
1012 12↵
1151 2↵
```

  - zapis binarne datoteke `artikli.bin` sadrži šifru artikla (short), naziv artikla (20+1 znak) i cijenu jednog komada artikla (float). Redni broj zapisa u datoteci odgovara šifri artikla. Napisati program koji će na zaslon ispisati račun u sljedećem obliku:

```
Telefon Kanasonic····12····10.00···120.00 kn↵
CD Player Suny·····2··1100.10··2200.20 kn↵
UKUPNO:·····················2320.20 kn↵
```

Rješenje (1. dio)

```
#include <stdio.h>

int main(void) {
   FILE *kup = NULL, *art = NULL;

   struct {
      short sifArt;
      char nazArt[20+1];
      float cijena;
   } artZapis;

   short sifArt, kolicina;
   float suma = 0;

   kup = fopen("kupljeno.txt", "r");
   art = fopen("artikli.bin", "rb");
```

Rješenje (2. dio)

```
   while (fscanf(kup, "%hd %hd", &sifArt, &kolicina) == 2) {
      fseek(art, (long)sizeof(artZapis) * (sifArt - 1), SEEK_SET);
      // procitaj cijeli zapis u strukturu
      fread(&artZapis, sizeof(artZapis), 1, art);
      // ispisi redak racuna na zaslon
      printf("%-20s %2d %8.2f %8.2f kn\n",
           artZapis.nazArt, kolicina,
           artZapis.cijena, artZapis.cijena * kolicina);
      suma += artZapis.cijena * kolicina;
   }
   printf("UKUPNO:%34.2f kn", suma);
   fclose(kup);
   fclose(art);
   return 0;
}
```

Primjer

- Programski zadatak
  - u tekstnu datoteku `ulaz.txt`, koja se nalazi u mapi `c:/tmp`, editorom su upisani podaci o osobama (matični broj i prezime). Prezime nije dulje od 15 znakova. Primjer sadržaja datoteke prikazan je ovdje:

```
952 Medvedec↵
101 Vurnek↵
205 Habajec↵
412 Voras↵
551 Ozimec↵
```

  - u novu tekstnu datoteku `izlaz.txt` u mapi `c:/tmp` prepisati podatke o osobama čije prezime sadrži slovo `r`. Primjer sadržaja datoteke prikazan je ovdje:

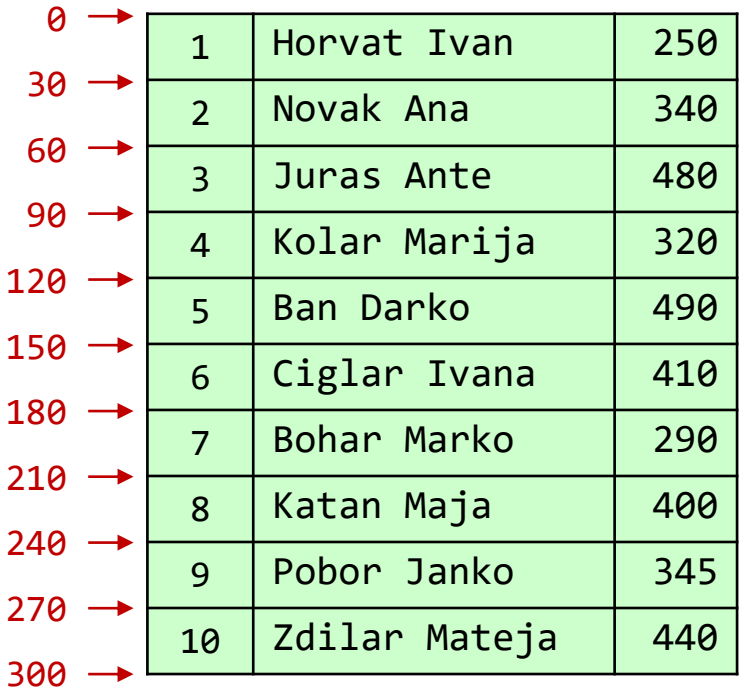
```
101 Vurnek↵
412 Voras↵
```

Rješenje

```
#include <stdio.h>
#include <string.h>

int main(void) {
   int mbr;
   char prez[15 + 1];

   FILE *ulTok = fopen("c:/tmp/ulaz.txt", "r");
   FILE *izTok = fopen("c:/tmp/izlaz.txt", "w");

   while (fscanf(ulTok, "%d %[^\n]", &mbr, prez) == 2) {
      if (strchr(prez, 'r') != NULL) {
         fprintf(izTok, "%d %s\n", mbr, prez);
      }
   }

   fclose (ulTok);
   fclose (izTok);

   return 0;
}
```

Primjer

- Programski zadatak
  - u binarnoj datoteci `bodovi.bin` nalaze se podaci o 10 studenata i bodovima koje su dobili na nekom predmetu. Svaki zapis sadrži matični broj (int), prezime i ime (21+1 znak) i broj bodova (int). Matični brojevi su u rasponu od 1-10, a redni broj zapisa odgovara matičnom broju.



- napisati program kojim će se za jednog slučajno odabranog studenta za 10% povećati dotadašnju vrijednost njegovih bodova. Ograničiti uvećani broj bodova na maksimalnih 500 bodova.

Rješenje (1. dio)

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

int main(void) {
   struct {
      int mbr;
      char prezIme[21+1];
      int brBod;
   } zapis;

   FILE *dUlIzl = fopen("bodovi.bin", "r+b");
   srand((unsigned)time(NULL));

   int mbr;
   mbr = rand() % 10 + 1;
```

čitanje **i** pisanje

Rješenje (2. dio)

```
   fseek(dUlIzl, (long)sizeof(zapis) * (mbr - 1), SEEK_SET);
   fread(&zapis, sizeof(zapis), 1, dUlIzl);
   // povecaj broj bodova (ali ne na vise od 500)
   zapis.brBod *= 1.1;
   if (zapis.brBod > 500)
      zapis.brBod = 500;
   // indikator pozicije vrati na pocetak zapisa!
   fseek(dUlIzl, -1L * sizeof(zapis), SEEK_CUR);
   // zapisi sadrzaj cijele strukture u datoteku
   fwrite(&zapis, sizeof(zapis), 1, dUlIzl);
   fclose(dUlIzl);
   return 0;
}
```

Ulazno/izlazni tok

- u prethodnom primjeru isti tok se koristio i za operacije čitanja i za operacije pisanja. Takav tok se naziva *ulazno/izlazni* tok
  - operacije čitanja i pisanja u ulazno/izlaznom toku smiju se koristiti naizmjence, ali
  - kada se u jednom toku nakon operacija čitanja žele početi obavljati operacije pisanja (ili obratno), tada se između operacija čitanja i operacija pisanja mora obaviti barem jedan poziv funkcije `fseek` ili barem jedan poziv funkcije `fflush`

Napomena

Prevodilac gcc na operacijskom sustavu Windows trenutačno po ovom pitanju još nije u potpunosti usklađen sa standardom: između operacija čitanja i pisanja (ili obrnuto) treba obaviti barem jedan poziv funkcije `fseek` (a ne po izboru ili `fseek` ili `fflush`).

fflush <stdio.h>

```
int fflush(FILE *stream);
```

- sadržaj međuspremnika toka zapisuje u sekundarnu memoriju (trajno pohranjuje)
  - radi efikasnog korištenja sekundarne memorije (koja je spora), pisanjem u tok podaci se prvo upisuju u međuspremnik u primarnoj memoriji, a tek zatim (u nekom kasnijem trenutku) se upisuju u sekundarnu memoriju (trajno pohranjuju)
  - to znači: ako program završi prije nego je obavljen `fflush` za neki tok, tada postoji mogućnost da neki od podataka koji jesu upisani u tok možda ipak neće biti zapisani u sekundarnu memoriju (i time će biti izgubljeni)
  - pozivom funkcije `fflush` osigurava se da su svi podaci do tog trenutka upisani u tok, također upisani i u sekundarnu memoriju
  - (funkcija `fclose` automatski obavlja `fflush` prije nego se tok zatvori)