Baze podataka I

Rad sa datotekama u programskom jeziku C

Datoteke

- datoteka
  - kao logička struktura podataka (LSP)
    - struktura nad skupom pojava jednog tipa entiteta
      - struktura slogova, nad datim tipom sloga
      - često se posmatra kao linearna struktura slogova
  - kao fizička struktura podataka (FSP)
    - predstavlja jednu LSP koja može biti viđena kao
      - linearna struktura (niz) slogova ili
      - niz znakova ili bajtova
    - smeštenu na eksterni memorijski uređaj
    - zajedno sa informacijama o samom načinu smeštanja LSP na uređaj

Datoteke

- vrste datoteka
  - binarne
    - sadržaj čine podaci predstavljeni u binarnom obliku
    - proizvoljan tip sadržaja
  - tekstualne
    - sadržaj čine podaci koji odgovaraju karakterima nekog kodnog sistema:
      - ASCII, EBCDIC, UTF-8, UTF-16
    - *plain-text*

Datoteke

- radnje nad datotekama
  - otvaranje i zatvaranje
  - pristup
    - čitanje, pisanje, pozicioniranje
  - provera statusa
  - kreiranje i brisanje

Datoteke

- pristup sadržaju datoteke (slogovima ili bajtovima)
  - sekvencijalan
    - pristupanje redosledom kojim je sadržaj raspoređen unutar datoteke
  - direktan
    - pristupanje delu sadržaja u proizvoljnom redosledu
      - na osnovu rednog broja
      - eksplicitnim pozicioniranjem (vrednost pokazivača)
  - pomoću ključa
    - pristupanja sadržaju u proizvoljnom redosledu
      - na osnovu dela sadržaja (ključa)

C

- podržane vrste datoteka
  - tekstualne datoteke
    - sa konverzijom sadržaja
  - binarne datoteke
    - pristup na nivou bajta
- podržane radnje nad datotekama
  - otvaranje, zatvaranje
  - čitanje, pisanje, pozicioniranje
  - provera statusa
  - kreiranje, brisanje

C

- pogled na datoteku
  - datoteka kao niz bajtova
    - bez složenije podele na slogove
      - neophodno dodatno programiranje
    - koncept toka - *stream*
- pristup sadržaju datoteke
  - sekvencijalan
    - bajt po bajt (podatak po podatak) po redosledu unutar datoteke
  - direktan
    - pozicioniranje na proizvoljan bajt

C - datotečki tip podatka

- tip podatka koji predstavlja datoteku u programu
  - FILE*
    - pokazivač na tip FILE
    - FILE je u osnovi struktura
      - deklarisana u zaglavlju stdio.h
    - primer deklaracije pokazivača datoteke:

```
FILE *f;
```

C stdio.h //MinGW

```
...
/*
 * The structure underlying the FILE type.
 *
 * Some believe that nobody in their right mind should make use of the
 * internals of this structure. Provided by Pedro A. Aranda Gutiirrez
 * <paag@tid.es>.
 */
#ifndef _FILE_DEFINED
#define _FILE_DEFINED
typedef struct _iobuf
{
        char*   _ptr;
        int     _cnt;
        char*   _base;
        int     _flag;
        int     _file;
        int     _charbuf;
        int     _bufsiz;
        char*   _tmpfname;
} FILE;
#endif /* Not _FILE_DEFINED */
...
```

C - datotečki tip podatka

- svaki ulaz i izlaz C programa se posmatra kao datoteka
- tri ugrađene datoteke
  - *stdin* - standardni ulaz
    - podrazumevano učitavanje vrednosti sa tastature
  - *stdout* - standardni izlaz
    - podrazumevano prikazivanje vrednosti na ekranu
  - *stderr* - standardni izlaz za poruke
    - poruke o greškama, prikazuje na ekranu
- infrastruktura za rad sa datotekama
  - `<stdio.h>`

C - otvaranje datoteke

- funkcija
- `FILE *fopen (const char *naziv, const char *rezim);`
  - `naziv` - naziv datoteke koja treba da bude otvorena
    - u skladu sa pravilima operativnog sistema
  - `rezim` - oznaka načina korišćenja datoteke
  - povratna vrednost
    - pokazivač na otvorenu datoteku ili
    - NULL vrednost ako otvaranje nije uspešno izvršeno
      - obavljati proveru povratne vrednosti

C - otvaranje datoteke

- mogući režimi rada - tekstualne datoteke
  - "r"
    - čitanje postojeće datoteke
  - "w"
    - pisanje u već postojećoj datoteci
      - prethodni sadržaj biva uništen
    - pisanje u novoj datoteci
      - automatski se kreira nova datoteka ako ne postoji neka sa datim nazivom
  - "a"
    - *append*
    - dodavanja sadržaja na kraj postojeće datoteke
    - ako datoteka sa datim nazivom ne postoji, biće kreirana

C - otvaranje datoteke

- mogući režimi rada - tekstualne datoteke
  - "r+" - čitanje i pisanje u postojećoj datoteci
  - "w+" - čitanje i pisanje
    - u postojećoj datoteci
      - prethodni sadržaj biva uništen
    - u novoj datoteci
      - automatski se kreira nova datoteka ako ne postoji neka sa datim nazivom
  - "a+"
    - čitanje i dodavanja sadržaja na kraj postojeće datoteke
    - ako datoteka sa datim nazivom ne postoji, biće kreirana
  - primer:
    - `FILE *f = fopen("sadrzaj.txt","r+");`

C - otvaranje datoteke

- mogući režimi rada - binarne datoteke
  - dodavanje slova *b* u opis režima
    - logika ostaje ista kao kod tekstualnih datoteka
  - moguće kombinacije
    - "rb", "wb", "ab"
    - "r+b", "w+b", "a+b"
      - "rb+", "wb+", "ab+" (identično kao prethodna tri)
  - primer:
    - `FILE *f = fopen("sadrzaj.bin","r+b");`

C - zatvaranje datoteke

- funkcija
- `int fclose(FILE *f);`
  - `f` - pokazivač na prethodno otvorenu datoteku koja treba da bude zatvorena
  - povratna vrednost
    - 0 ako je zatvaranje uspešno
    - konstanta EOF ako je došlo do greške
- automatsko zatvaranje
  - pri završetku izvršavanja programa
    - kraj *main* funkcije
    - sve otvorene datoteke bivaju zatvorene
  - ne oslanjati se na to
    - eksplicitno zatvoriti svaku otvorenu datoteku

C - primer 1

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

int main()
{
    FILE *f;
    char *n="sadrzaj.txt";

    if( (f=fopen(n,"r")) == NULL ){
        printf("Datoteka <%s> nije uspesno otvorena.\n", n);
        exit(1);
    }

    printf("Datoteka <%s> je uspesno otvorena.\n", n);
    printf("Zatvaranje datoteke <%s>...\n", n);

    if( fclose(f)==EOF )
        printf("\tNastupila je greska tokom zatvaranja!\n");
    else
        printf("\tZatvaranje uspesno!\n");

    return 0;
}
```

C - rad sa txt datotekama

- čitanje i pisanje
  - obavlja se sekvencijalno
    - od početka datoteke
  - svaki sledeći pristup
    - iza poslednjeg mesta kojem je pristupano u prethodnom čitanju ili pisanju

C - rad sa txt datotekama

- rad sa znakovima - čitanje
- `int fgetc(FILE *f);`
  - čitanje pojedinačnog znaka
  - `f` - pokazivač na prethodno otvorenu datoteku
  - povratna vrednost
    - kod procitanog znaka
    - konstanta EOF ako se došlo do kraja datoteke ili se desila neka greška
- `char *fgets(char *tekst, int n, FILE *f);`
  - čitanje teksta iz datoteke
    - do znaka '\n' ili
    - ukupno n-1 znakova
  - upisuje '\0' na kraj stringa
  - `tekst` - string koji prihvata učitani sadržaj
  - `n` - maksimalni broj znakova za učitavanje (uključujući '\0')
  - `f` - pokazivač na prethodno otvorenu datoteku
  - povratna vrednost
    - adresa na koju je upisan novi sadržaj - adresa stringa `tekst`
    - konstanta NULL ako se došlo do kraja datoteke ili se desila neka greška

C - rad sa txt datotekama

- rad sa znakovima - pisanje
- `int fputc(int c, FILE *f);`
  - upisivanje pojedinačnog znaka
  - `c` - kod znaka za upis
  - `f` - pokazivač na prethodno otvorenu datoteku
  - povratna vrednost
    - kod upisanog znaka
    - konstanta EOF ako se desila neka greška
- `int fputs(const char *tekst, FILE *f);`
  - upisivanje stringa u formi jednog reda datoteke
    - znak '\n' automatski dodaje na kraj
  - `tekst` - string koji predstavlja sadržaj za upis
  - `f` - pokazivač na prethodno otvorenu datoteku
  - povratna vrednost
    - ne-negativna vrednost
    - konstanta EOF ako se desila neka greška

C - rad sa txt datotekama

- rad sa znakovima - ulazna i izlazna konverzija
- `int fscanf(FILE *f, const char *form, a1, a2, ...);`
  - učitavanje, izvršenje konverzije za a1, a2 ... prema zadatom formatu
  - `f` - pokazivač na prethodno otvorenu datoteku
  - `form` - specifikacija konverzije
  - `a1, a2, ...` - adrese za smeštanje sadržaja učitanog iz datoteke
  - povratna vrednost
    - broj uspešno konvertovanih i smeštenih podataka
    - konstanta EOF ako se pojavio kraj datoteke ili desila neka greška pre prve konverzije
- `int fprintf(FILE *f, const char *form, a1, a2, ...);`
  - upisivanje, izvršenje konverzije za a1, a2 ... prema zadatom formatu
  - `f` - pokazivač na prethodno otvorenu datoteku
  - `form` - specifikacija konverzije
  - `a1, a2, ...` - adrese za čitanje sadržaja za upis u datoteku
  - povratna vrednost
    - broj ispisanih znakova
    - negativna vrednost (-1) ako se desila neka greška pri konverziji

C – primer 2

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

int main()
{
    FILE *f;
    char *n="test.txt";
    char c;

    if( (f=fopen(n,"r")) == NULL ){
        printf("Datoteka <%s> nije uspesno otvorena.\n", n);
        exit(1);
    }

    printf("Datoteka <%s> je uspesno otvorena.\n", n);
    printf(">>Tekstualni sadrzaj datoteke <%s>...\n", n);

    while((c=fgetc(f))!=EOF)
        putchar(c);

    printf("\n>>Kraj sadrzaja datoteke <%s>...\n", n);
    fclose(f);

    return 0;
}
```

C - zadatak 1

- učitati niz od *n* celih brojeva sa tastature
  - pri čemu je korisnik prethodno zadao *n* isto preko tastature
- kreirati tekstualnu datoteku sa nazivom *naz* koja će u svakom redu sadržati po jedan učitani broj, razmak i vrednost tog broja dignutu na drugi stepen
  - pri čemu je korisnik prethodno zadao *naz* preko tastature
  - primer
    - za niz {3,5,7} datoteka može izgledati ovako

```
3 9
5 25
7 49
```

C - rad sa bin datotekama

- čitanje i pisanje
  - obavlja se sekvencijalno
    - od početka datoteke
    - svaki sledeći pristup
      - iza poslednjeg mesta kojem je pristupano u prethodnom čitanju ili pisanju
  - mogućnost pozicioniranja
    - zadavanje pomeraja u odnosu na referentnu tačku
      - početak, trenutna pozicija, kraj datoteke
    - sledeći pristup od poslednje postavljene pozicije
  - programer vodi računa o strukturi datoteke

C - rad sa bin datotekama

- rad sa podacima - ulaz
- `int fread(void *sadrzaj, int duz, int n, FILE *f);`
  - učitavanje zadatog broja podataka u memoriju iz datoteke od prethodne pozicije
  - nova pozicija u datoteci na mestu iza poslednjeg učitanog bajta
  - `sadrzaj` - memorijska adresa od koje počinje smeštanje učitanih podataka
  - `duz` - dužina pojedinačnog podatka za učitavanje (u bajtovima)
  - `n` - broj podataka koji treba učitati iz datoteke
  - `f` - pokazivač na prethodno otvorenu datoteku
  - povratna vrednost
    - broj uspešno pročitanih podataka
  - indikator greške učitavanja
    - `feof(), ferror()`

C - rad sa bin datotekama

- rad sa podacima - izlaz
- `int fwrite(const void *sadrzaj, int duz, int n, FILE *f);`
  - upisivanje zadatog broja podataka zadate dužine iz memorije u datoteku od mesta završetka poslednjeg pristupa
  - ako pozicija nije kraj datoteke, dolazi do prepisivanja sadržaja
  - ako kapacitet datoteke nije dovoljan, datoteka se povećava
  - nova pozicija u datoteci na mestu iza poslednjeg upisanog bajta
  - `sadrzaj` - memorijska adresa od koje počinje čitanje podataka za smeštanje u datoteku
  - `duz` - dužina pojedinačnog podatka za smeštanje (u bajtovima)
  - `n` - broj podataka koji treba smestiti u datoteku
  - `f` - pokazivač na prethodno otvorenu datoteku
  - povratna vrednost
    - broj uspešno upisanih podataka
    - ako je došlo greške onda je ta vrednost manja od `n`

C - rad sa bin datotekama

- rad sa podacima - pozicioniranje
- `int fseek(FILE *f, long ofset, int ref);`
  - postavljanje trenutne pozicije unutar datoteke
  - može i za tekstualne datoteke
  - neophodna kada posle radnje čitanja dolazi pisanje (i obrnuto)
  - `f` - pokazivač na prethodno otvorenu datoteku
  - `ofset` - udaljenost nove pozicije od referentne tačke (u bajtovima)
  - `ref` - referentna tačka u odnosu na koju se posmatra udaljenost nove pozicije
    - `SEEK_SET` - početak datoteke
    - `SEEK_CUR` - trenutna pozicija unutar datoteke
    - `SEEK_END` - kraj datoteke
  - povratna vrednost
    - oznaka greške - ako vrednost različita od 0

C - rad sa bin datotekama

- rad sa podacima - pozicioniranje
- `long ftell(FILE *f);`
  - nalaženje trenutne pozicije unutar datoteke
  - `f` - pokazivač na prethodno otvorenu datoteku
  - povratna vrednost
    - trenutna pozicija u bajtovima u odnosu na početak datoteke
    - `-1L` - oznaka greške
- `void rewind(FILE *f);`
  - postavljanje pozicije na početak datoteke
  - `f` - pokazivač na prethodno otvorenu datoteku
  - nema povratnu vrednost

C - rad sa datotekama

- indikacija greške
  - vrednosti indikatora se postavljaju kroz bočne efekte mnogih prethodnih funkcija
  - dva indikatora
    - indikator kraja datoteke
    - indikator greške
  - funkcije
  - `void clearerr(FILE *f);`
    - briše vrednosti oba indikatora
    - `f` - pokazivač na prethodno otvorenu datoteku
  - `int feof(FILE *f);`
    - provera indikatora kraja datoteke
    - `f` - pokazivač na prethodno otvorenu datoteku
    - povratna vrednost različita od 0 ako je indikator uključen
  - `int ferror(FILE *f);`
    - provera indikatora greške za datoteku
    - `f` - pokazivač na prethodno otvorenu datoteku
    - povratna vrednost različita od 0 ako je indikator uključen
  - globalna promenljiva - `errno` u `<errho.h>` - kod greške

C – primer 3

```
#include <stdio.h>
#include <stdlib.h>
#define ROWCOUNT 16

int main()
{
    FILE *f;
    char c, *n="test.txt";
    long cnt=0;

    if( (f=fopen(n,"rb")) == NULL ){
        printf("Datoteka <%s> nije uspesno otvorena.\n", n);
        exit(1);
    }
    printf(">>Binarni sadrzaj datoteke <%s>...\n", n);

    while(fread(&c,sizeof(char),1,f)){

        if(cnt%ROWCOUNT==0)
            printf("\n");

        printf("0x%02x ",c);
        cnt++;
    }
    printf("\n>>Kraj sadrzaja datoteke <%s>...\n", n);
    fclose(f);
    return 0;
}
```

C - zadatak 2

- učitati niz od *n* celih brojeva sa tastature
  - pri čemu je korisnik prethodno zadao *n* isto preko tastature
- kreirati binarnu datoteku sa nazivom *naz* koja će za svaki učitani broj sadržati njegovu vrednost kao i tu vrednost dignutu na drugi stepen
  - pri čemu je korisnik prethodno zadao *naz* preko tastature
  - primer
    - za niz {3,5,7} datoteka može izgledati ovako -*little endian*

```
03 00 00 00 09 00 00 00
05 00 00 00 19 00 00 00
07 00 00 00 31 00 00 00
```

C - zadatak 3

- za datoteku čiji je naziv uneo korisnik preko tastature
  - prikazati veličinu datoteke
    - u B, kB i MB
    - oslanjajući se na prethodne funkcije
  - prikazati odgovarajuću poruku u slučaju greške prilikom nalaženja i otvaranja datoteke

C - zadatak 4

- omogućiti korisniku da bira jednu od četiri opcije u radu sa binarnom datotekom
  - otvaranje datoteke sa nazivom koji korisnik unese
    - kao i zatvaranje prethodne a otvaranje nove datoteke
  - prikaz bajta iz otvorene datoteke pri čemu korisnik zadaje rednu poziciju sa koje želi da učita bajt
  - izmenu bajta u otvorenoj datoteci pri čemu korisnik zadaje rednu poziciju kao i vrednost koja treba da se upiše u datoteku na toj poziciji
  - kraj rada sa programom
    - kao i zatvaranje otvorene datoteke

C - zadatak 5

- omogućiti korisniku da bira jednu od tri opcije
  - unos koordinata tačke u 3d prostoru i upis te tačke u datoteku *tacke.bin* (realizovati kao posebnu funkciju)
    - koordinata je tipa *double*
  - automatsko računanje obima najmanje sfere koja obuhvata sve tačke koje su sačuvane u *tacke.bin* (realizovati kao posebnu funkciju)
    - sa prolazom kroz datoteku radi čitanja koordinata tačaka
  - kraj rada sa programom
    - kao i zatvaranje datoteke

C - zadatak 5-2

- za prethodni program
  - napisati funkciju koja će omogućiti korisniku da prikaže koordinate *i*-te tačke iz datoteke
  - napisati funkciju koja će omogućiti korisniku da upiše u datoteku nove vrednosti koordinata *i*-te tačke
    - umesto postojećih vrednosti
  - napisati funkciju za brisanje *i*-te tačke iz datoteke
    - realizovati kao logičko brisanje
  - dodati prethodne opcije u korisnički meni
  - prilagoditi funkcije da podrže logičko brisanje