Programiranje 2
Programski jezik C

Datoteke,
Argumenti komandne linije

Ana Spasić
14. februar 2011.

Datoteke

- U C- u se sa datotekama barata na sledeći način:

  u standardnoj biblioteci postoji definisana struktura:

  ```
  typedef struct {
    ....
  } FILE;
  ```

  koja sadrži sve informacije vezane za otvorenu datoteku. Ove informacije su platformski zavisne, i nama nisu od značaja.

- Njima barataju funkcije standardne biblioteke.

Datoteke

- Datoteka se otvara funkcijom:

  FILE * fopen(char *ime, char *nacin);

  Ova funkcija prihvata string sa putanjom do datoteke (ime), kao i string (nacin) koji definiše način rada sa datotekom. Može biti:

  "r" – datoteka se otvara za čitanje (mora da postoji)

  "w" – datoteka se otvara za pisanje (kreira se ako ne postoji, a ako postoji, njegov sadržaj se briše)

  "a" – datoteka se otvara za dopisivanje (kreira se ako ne postoji. Prilikom svakog pisanja sadržaj se dodaje na kraj datoteke).

Datoteke

- Funkcija **fopen()** vraća pokazivač na FILE strukturu koja je povezana sa otvorenom datotekom. Od tog trenutka se baratanje sa otvorenom datotekom sastoji u tome da se funkcijama standardne biblioteke prosledi ovaj pokazivač kao argument. Ostalo obavlja biblioteka za nas.
- Ako datoteka nije uspešno otvorena, povratna vrednost f-je je **NULL**. Ova konstanta je definisana u stdio.h zaglavlju na sledeći način:

```
#define NULL 0
```

Dakle, u pitanju je nula. Uobičajeno je da se ova konstanta koristi za pokazivače koji imaju vrednost nula (tj. ne pokazuju ni na šta).

Datoteke

- Prilikom svakog otvaranja datoteke, treba proveriti da li je nastupila greška, tj. da li je povratna vrednost funkcije bila NULL. Ako jeste, treba nekako obraditi tu grešku. Najčešći način za to je obavestiti korisnika o tome šta se desilo i prekinuti program.

- Nakon završetka rada sa datotekom, treba ga zatvoriti sa: **fclose(f);** gde je f FILE pokazivač,

  jer operativni sistem ograničava broj datoteka koje program istovremeno može da drži otvorenim.

- Prilikom završetka rada programa, sve datoteke se automatski zatvaraju. Nećemo se oslanjati na ovo, već kad god nam otvorena datoteka više ne treba, eksplicitno ćemo je zatvarati.

Datoteke

- Standardni ulaz i izlaz se u C – u takođe svodi na ulaz i izlaz iz datoteka. Zapravo, prilikom pokretanja programa automatski se otvaraju tri datoteke, kojima se dodeljuju sledeći globalni FILE pokazivači:
- **stdin –** standardni ulaz (povezan sa tastaturom)
- **stdout** – standardni izlaz (povezan sa ekranom)
- **stderr –** standardni izlaz za greske (povezan sa ekranom)

- **stdin** i **stdout** se mogu preusmeriti koristeći <, odnosno > u komandnoj liniji, dok **stderr** ne može (tj. ostaje uvek povezan sa ekranom)

Čitanje i pisanje -karakter po karakter-

- Funkcije iz standardne biblioteke:

  int fgetc(FILE *f);

  Učitava jedan karakter iz datoteke na koji pokazuje f, i vraća njegov ASCII kod kao povratnu vrednost. Ako je nastupio kraj datoteke, vraća EOF.

  int fputc(int c, FILE *f);

  Upisuje karakter c u datoteku na koji pokazuje f. Vraća c.

- Imajući ovo u vidu, lako je uočiti da su sledeće naredbe ekvivalentne:

  getchar() <=> fgetc(stdin)

  putchar(c) <=> fputc(c, stdout);

Čitanje i pisanje -linija po linija-

- **char *fgets(char *s, int limit, FILE *f);**

  Učitava celu liniju karaktera iz datoteke na koji pokazuje f u string s, ali ne više od limit-1 karaktera iz datoteke (limit treba da bude veličina stringa s). Eventualni '\n' sa kraja linije se čuva u stringu s, koji se ovom f-jom i terminira nulom. F-ja vraća NULL ako se desila greška pri učitavanju, inače vraća s.

- **int fputs(const char *s, FILE *f);**

  Upisuje string s u datoteku na koji pokazuje f bez terminirajuće nule.

  Vraća EOF za grešku.

Formatizovano čitanje i pisanje

- **int fscanf(FILE *f, const char *format, ...);**
  int fprintf(FILE *f, const char *format, ...);
- Ove dve funkcije rade isto što i scanf i printf, osim toga što ne rade sa standardnim ulazom, tj. sa standardnim izlazom već rade sa datotekom na koju pokazuje f.
- Povratna vrednost je broj uspešnih konverzija. Ukoliko dođe do kraja datoteke pre prvog pročitanog podatka ili u slučaju greške povratna vrednost biće EOF.

Datoteke

- **int feof(FILE *f)**

  F-ja provera da li je datoteka kojoj odgovara pokazivač f došla do kraja. Ukoliko se stiglo do kraja datoteke vraća vrednost različitu od 0.

- **long ftell(FILE *f);**

  Ova f-ja vraća tekuću poziciju datoteke f ili 1L u slučaju greške.

- Ako nam zatrebaju detaljniji opis neke od ovih funkcija možemo ga naći u man stranicama za tu funkciju. U konzoli kucamo

  man fprintf

  ako tražimo dodatni opis funkcije fprintf.

Datoteke

- **int fseek(FILE *f, long offset, int origin);**

  Funkcija fseek podešava poziciju datoteke f. Narednim čitanjem ili pisanjem pristupa se podacima na početku nove pozicije. Nova pozicija se postavlja na offset znakova od mesta origin, koje može biti **SEEK_SET** (početak datoteke), **SEEK_CUR** (tekuća pozicija) ili **SEEK_END** (kraj datoteke).

  Funkcija fseek vraća vrednost različitu od nule u slučaju greške.

- Primer: fseek(f, 0, SEEK_END) postavlja datoteku na 0 bajtova od kraja datoteke, tj. na kraj datoteke.

Datoteke – obrada greške

- Stderr je standardni izlaz za greške. Povezan je sa ekranom. Ne može se preusmeriti. Za ispis poruke o grešci na stderr koristimo f-ju fprintf

  fprintf(stderr, "Greska: program pri čitanju.\n");

- Kada se dogodi greška posle koje naš program ne može da nastavi sa izvršavanjem potrebno je i prekinuti program pozivom f-je deklarisane u stdlib.h

  void exit(int status)

- Uspešno izvršavanje exit(EXIT_SUCCESS);

  Neuspelo izvršavanje programa exit(EXIT_FAILURE);

- Gde su EXIT_SUCCESS i EXIT_FAILURE konstante definisane u stdlib.h.

Argumenti komandne linije

- Prilikom pozivanja programa iz komandne linije moguće je navesti određene opcije i argumente, koje program koristi u svom radu. Npr, možemo pozvati program na sledeći način:

  ./izvrsni ulaz.txt izlaz.txt

  zeleći da program izvrsni iskopira datoteku ulaz.txt u datoteku izlaz.txt.
- Potrebno je da postoji mehanizam da program iznutra pristupi stringovima koje je korisnik naveo prilikom poziva programa.
- U C-u se to ostvaruje na sledeći način:

Argumenti komandne linije

- Neophodno je da fja main ima 2 parametra, i to:

  int main(int argc, char **argv)

- argc i argv su parametri koji se programu prenose prilikom pokretanja i tada dobijaju sledeće vrednosti:
- **argc** – broj argumenata navedenih u komandnoj liniji prilikom poziva programa.

  I ime programa broji kao argument komandne linije. U primeru će argc biti postavljeno na 3.
- **argv** – niz stringova koji sadrži argumente navedene u komandnoj liniji pri pozivu programa.

Argumenti komandne linije

- argv[0] je uvek ime pod kojim je program pozvan. Stoga i u slučaju f-je int main() imamo 1 podrazumevani argument komandne linije koji je ime programa.

- Npr. u gornjem primeru, argv[0] je string "./primer", argv[1] je string "ulaz.txt", argv[2] je string "izlaz.txt".
- Dakle, niz pokazivača dužine upravo argc, na čiji prvi član pokazuje argv.

Argumenti komandne linije

- Uobičajena praksa na UNIX sistemima je da se argumenti komandne linije dele na **opcije** i **argumente u užem smislu** (kraće argumenti).
- Opcije počinju znakom '-' nakon čega obično sledi jedan karakter koji označava koja je opcija u pitanju. Ovim se najčešće upravlja funkcionisanjem programa i neke mogućnosti se uključuju ili isključuju. Argumenti najčešće predstavljaju opisne informacije poput imena datoteke, ili podatke nekog drugog tipa.

gcc –o izvrsni primer.c

- U prethodnom pozivanju prevodioca, -o je opcija, a izvrsni i primer.c su argumenti u užem smislu.

Zadaci

1. Napisati program koji prepisuje datoteku ulaz.txt u datoteku izlaz.txt i to (a) karakter po karakter (b) liniju po liniju.
2. Napisati program koji ispisuje broj navedenih argumenata komandne linije, a zatim i same argumenate sa rednim brojevima.
3. Napisati program koji prepisuje datoteku čije se ime navodi kao prvi argument komandne linije u datoteku čiji se argument navodi kao drugi argument komandne linije programa. Ukoliko je navedena opcija u program prilikom prepisivanja treba da zamenjuje sva mala slova velikim, a ukoliko je navedena opcija l sva velika slova se zamenjuju malim. Ako nisu navedene opcije izvršiti samo prepisivanje iz jedne u drugu datoteku.