Spisak zadataka za vežbanje

Napomena: kod prevoditi sa opcijom –std=c99.

1. Napisati program koji kao argumente komandne linije prima listu regularnih fajlova koji sadrže samo znakove `"+"`, `"-"` i beline.

   Za svaki fajl pokrenuti posebnu nit u kojoj se taj fajl obrađuje (i-ta nit obrađuje i-ti fajl). Svaka nit treba da izračuna skor za fajl, pri čemu se skor računa kao razlika broja `"+"` i `"-"` karaktera. Dodatno, svaka nit mora da ažurira globalni skor tako što će ga uvećati za izračunati lokalni skor.

   U main() funkciji ispisati vrednost lokalnog skora za sve niti redom (vratiti ovu vrednost kao povratnu vrednost funkcije koju nit izvršava) i na kraju vrednost globalnog skora (svaki broj u svom redu).

2. Napisati program koji pokreće komandnu `ls -l` u odvojenom procesu, preusmerava standardni izlaz te komande u `PIPE` i filtrira ovaj sadržaj tako da ispisuje samo petu kolonu izlaza ove komande (to je veličina fajla).

3. Napisati program koji prima kao argument komandne linije putanju do C fajla. Za prosleđeni C fajl, program poziva gcc kompajler sa opcijom -S, čeka na završetak prevođenja i ispisuje broj mov komandi u dobijenom .s fajlu. Na primer, poziv komande može biti `gcc -std=c99 -S 2.c -o 2.s`. Radi jednostavnosti, pretpostaviti da je broj mov komandi u stvari broj linija u .s fajlu u kojima se `"mov"` pojavljuje kao podstring.

   Ukoliko se gcc ne izvrši uspešno, program treba da završi sa exit code-om sa kojim je gcc završio izvršavanje.

   Po završetku rada program treba da obriše pomoćni .s fajl.

   Dozvoljeno je koristiti funkcije iz zaglavlja string.h.

4. Napisati program sa više niti koji računa sumu matrice koja se dobija tačka po tačka proizvodom dve kvadratne matrice (eng. dot product). Kao argument komandne linije program prima putanju do fajla u kome se nalaze matrice. Prvi broj u ovom fajlu predstavlja dimenziju kvadratne matrice N, a zatim sledi $2 * N^2$ realnih brojeva koji su elementi matrica (prvih N^2 za prvu matricu, a drugih za drugu). Potrebno je pokrenuti N niti tako da jedna nit računa sumu tačka po tačka proizvoda za jedan par redova. Koristiti mutekse za sinhronizaciju. Ukupnu sumu ispisati iz main() funkcije.

5. Napisati program koji u beskonačnoj petlji učitava cele pozitivne brojeve. Nakon učitavanja broja program čeka na signal. U zavisnosti od toga koji signal je stigao, program treba da ispiše modifikovanu verziju broja.

   - `SIGUSR1` - ispisuje se kvadrat broja
   - `SIGUSR2` - ispisuje se kub broja
   - `SIGINT` - ispisuje se obrnut broj (npr. za 123 ispisuje se 321)
   - `SIGQUIT` - izlazi se iz programa bez obrade poslednje učitanog broja.

   Osim brojeva ne ispisivati dodatan tekst. Smatrati da naredni signal neće biti poslat dok se ne završi obrada tekućeg broja.

6. Napisati program koji kao argument komandne linije dobija broj sekundi proteklih od Epohe, a kroz promenljivu okruženja `PUTANJA` dobija putanju do fajla. Program u beskonačnoj petlji čeka na signal.

   - Ukoliko program dobije signal `SIGUSR1`, na standardni izlaz ispisuju se godina poslednjeg pristupa i godina poslednje modifikacije prosleđenog fajla.
   - Ukoliko program dobije signal `SIGUSR2`, postaviti vreme pristupa i vreme modifikacije prosleđenog fajla na broj sekundi koji se zadaje kao argument komandne linije.
   - Ako program dobije signal `SIGQUIT` završava se čekanje na signale i izlazi iz petlje.

7. Napisati program koji u fajlu koji se dobija kao argument komandne linije bezbedno menja sva pojavljivanja reči `"Milivoje"` sa `"Dragutin"`. Pod bezbednim menjanjem smatra se da se deo fajla gde je pronađena reč `"Milivoje"` zaključa za pisanje, zatim se promeni sadržaj fajla sa `"Milivoje"` na `"Dragutin"` i na kraju se otključa katanac (postupak ponoviti za svako pojavljivanje).

   Ukoliko nije moguće zaključati fajl, ispisati jedno slovo i dva broja:

   - slovo `w` ili `r` u zavisnosti od toga da li katanac koji neki drugi proces drži za pisanje ili čitanje,
   - poziciju u fajlu gde počinje reč koju nije bilo moguće zaključati,

- identifikator procesa koji drži konfliktni katanac.

Pomoć: Kada pročitate reč Milivoje koristite fseek() da se vratite unazad u fajlu.

8. Promenljiva okruženja PATH sadrži spisak putanja do različitih direktorijuma u formatu:

```
PATH=«dirpath_1>:<dirpath_2>:...:<dirpath_n>"
```

Napisati program koji formira novu promenljivu `PATH_FILTERED` koja sadrži samo putanje do direktorijuma iz promenljive `PATH` za koje grupa ima pravo pisanja. Novoformiranu promenljivu dodati u okruženje.

Napomena: na računarima na kojima se polaže ispit sadržaj promenljive `PATH` se može razlikovati od onog u primeru koji dobijete uz zadatak. Sadržaj promenljive `PATH` možete videti u terminalu pomoću komande `echo $PATH`.