Elektrotehnički fakultet u Beogradu
Katedra za računarsku tehniku i informatiku

Predmet: Operativni sistemi 2 (SI3OS2, IR3OS2)
Nastavnik: prof. dr Dragan Milićev
Odsek: Softversko inženjerstvo, Računarska tehnika i informatika
Kolokvijum: Treći, januar 2016.
Datum: 12.1.2016.

Treći kolokvijum iz Operativnih sistema 2

Kandidat: _____

Broj indeksa: _____ *E-mail:*_____

Kolokvijum traje 1,5 sat. Dozvoljeno je korišćenje literature.

Zadatak 1 _____/10 *Zadatak 3* _____/10
Zadatak 2 _____/10

Ukupno: _____/30 = _____%

Napomena: Ukoliko u zadatku nešto nije dovoljno precizno definisano, student treba da uvede razumnu pretpostavku, da je uokviri (da bi se lakše prepoznala prilikom ocenjivanja) i da nastavi da izgrađuje preostali deo svog odgovora na temeljima uvedene pretpostavke. Ocenjivanje unutar potpitanja je po sistemu "sve ili ništa", odnosno nema parcijalnih poena. Kod pitanja koja imaju ponuđene odgovore treba **samo zaokružiti** jedan odgovor. Na ostala pitanja odgovarati **čitko, kratko i precizno**.

1. (10 poena) Upravljanje diskovima

U nekom sistemu klasa `DiskScheduler`, čiji je interfejs dat dole, realizuje raspoređivač zahteva za pristup disku po *C-LOOK* algoritmu. Operacija `put()` smešta novi zahtev za pristup disku u red čekanja, a operacija `get()` izbacuje iz reda i vraća zahtev koji je sledeći na redu za opsluživanje. U strukturi `DiskRequest` polje `cyl` tipa `unsigned int` označava broj cilindra na koji se zahtev odnosi, a polja `prev` i `next` predstavljaju pokazivače na prethodni i sledeći zahtev u dvostruko ulančanoj listi zahteva koju održava `DiskScheduler`.

Implementirati celu klasu `DiskScheduler`.

```
class DiskScheduler {
public:
  DiskScheduler ();
  DiskRequest* get ();
  void put (DiskRequest*);
};
```

Rešenje:

2. (10 poena) Operativni sistem Linux

Napisati *shell* skriptu koji ispisuje vreme najskorijeg polaska voza. Skripta ima četiri argumenta. Prvi parametar je ime fajla u kojem se nalazi red vožnje, drugi je redni broj voza za koji se vrši pretraga, dok treći i četvrti parametar predstavljaju trenutno vreme, gde je treći parametar oznaka sata, a četvrti minuta. U slučaju neodgovarajućeg broja parametara ili neispravnog prvog parametra, ispisati odgovarajuću poruku i prekinuti skriptu. U redu vožnje svaki red sadrži informacije o vremenu polaska za jedan voz. Na početku reda se nalazi redni broj voza, a nakon rednog broja su zapisana vremena polazaka u formatu hh:mm, koja su sortirana po rastućem poretku. Skripta treba da za zadati voz ispiše vreme najskorijeg polaska tog voza u odnosu na trenutno vreme.

Primer jednog reda fajla `red.txt` je:
1001 14:15 14:30 14:50

Primer izlaza jednog poziva skripta sa argumentima `red.txt 1001 14 29` je:
14:30

Rešenje:

3. (10 poena) Operativni sistem Linux

Na jeziku C/C++, koristeći mehanizam razmene poruka kod operativnog sistema Linux, napisati funkciju za sinhronizaciju više procesa. Nije potrebno proveravati uspešnost sistemskih poziva za slanje i primanje poruka. Telo jednog procesa dato je u nastavku. Dozvoljeno je koristiti samo jedno sanduče za komunikaciju. Broj procesa koji se pokreće dat je kao konstanta NUM_PROCESS. Procesi se pokreću sa jednim argumentom koji predstavlja redni broj procesa. Redni brojevi su od 0 do NUM_PROCESS – 1.

Sinhronizacija se vrši pomoću funkcije:

```
void barrier(int id, int msg_box);
```

gde je parametar `id` redni broj procesa, dok je parametar `msg_box` ručka za pristup sandučetu za razmenu poruka preko koga se vrši komunikacija među procesima. Proces koji pozove ovu funkciju treba da bude blokiran sve dok svi ostali procesi ne pozovu istu funkciju.

Potrebno je implementirati funkciju `barrier` i definisati strukturu `msgbuf`.

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/msg.h>
#include <unistd.h>

const int BOX_KEY = …;
const int NUM_PROCESS = …;

struct msgbuf;
void barrier(int id, int msg_box);

int main(int argc, char *argv[])
{
  if (argc != 2) {
    fprintf(stderr, "Nedovoljno argumenata.\n");
    return 1;
  }
  int id = atoi(argv[1]);

  int msg_box = msgget(BOX_KEY, IPC_CREAT | 0666);

  while (…) {
    /* Work */
    barrier(id, msg_box);
  }

  msgctl(msg_box, IPC_RMID, 0);

  return 0;
}
```

Rešenje: