

---

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:* Prvi, septembar 2014.

*Datum:* 19.8.2014.

*Prvi 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) Raspoređivanje procesa

U nekom sistemu klasa `Scheduler`, čiji je interfejs dat dole, realizuje raspoređivač spremnih procesa po prioritetu (engl. *priority scheduling*), tako da i operacija dodavanja novog spremnog procesa `put()` i operacija uzimanja spremnog procesa koji je na redu za izvršavanje `get()` imaju ograničeno vreme izvršavanja koje ne zavisi od broja spremnih procesa (kompleksnost  $O(1)$ ). Važe sledeće pretpostavke:

- Svi prioriteti su celi brojevi u opsegu  $0 \dots \text{MAXPRI}$ . Veća vredost znači viši prioritet.
- Svaki proces ima svoj podrazumevani prioritet (polje `defaultPri` u `PCB`) koji je fiksna (zadaje se pri kreiranju procesa) i dinamički prioritet (polje `priority` u `PCB`) koji se menja tokom izvršavanja, ali samo u opsegu `defaultPri ± PRIRANGE`.
- Dinamička promena prioriteta vrši se prema uobičajenim pravilima za interaktivne sisteme, prilikom smeštanja procesa u listu spremnih. Svaka promena dinamičkog prioriteta je u inkrementima `PRISTEP`.
- U operaciji `put()` drugi argument ukazuje na to da li proces dolazi u listu spremnih iz stanja suspenzije (`wasBlocked==1`), ili iz stanja spremnosti, pošto mu je isteklo dodeljeno vreme izvršavanja (`wasBlocked==0`).
- Između spremnih procesa sa istim prioritetom raspoređivanje je *FCFS*.
- U slučaju da nema drugih spremnih procesa, treba vratiti proces na koga ukazuje `idle` (to je uvek spreman proces najnižeg prioriteta).
- U strukturi `PCB` polje `next` pokazivač tipa `PCB*` koji služi za ulančavanje struktura `PCB` u jednostruke liste.

Realizovati u potpunosti klasu `Scheduler`.

```
const int MAXPRI = ..., PRIRANGE = ..., PRISTEP = 1;
extern PCB* idle;

class Scheduler {
public:
    Scheduler ();
    PCB* get ();
    void put (PCB* pcb, int wasBlocked);
};
```

Rešenje:

**2. (10 poena) Međuprocesna komunikacija pomoću deljene promenljive**

Korišćenjem klasičnih uslovnih promenljivih, napisati kod za monitor koji realizuje binarni semafor (dogadjaj).

Rešenje:

### 3. (10 poena) Međuprocena komunikacija razmenom poruka

Na programskom jeziku Java napisati program koji obezbeđuje sinhronizaciju prolaska vozila kroz kružni tok. U jednom trenutku u kružnom toku najviše se može nalaziti  $N$  vozila. U kružni tok se uliva  $M$  ulica, koje su označene rednim brojem  $0..M-1$ . Svako vozilo predstavljeno je jednim klijentskim procesom. Prilikom zahteva za ulazak u kružni tok vozilo prosleđuje serverskom procesu i redni broj ulice iz koje dolazi. U slučaju da više vozila čeka na ulazak u kružni tok, vozila se puštaju po redosledu rednog broja ulice iz koje dolaze. Po izlasku iz kružnog toka vozilo to prijavljuje serverskom procesu. Serverski proces „osluškuje“ port 1033 preko koga prima zahteve. Međuprocenu komunikaciju realizovati preko priključnica (*socket*) i razmenom poruka (*message passing*). Nije potrebno proveravati uspešnost izvršavanja operacija (*try/catch* klauzule).

Rešenje: