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, oktobar 2011.
Datum: 30.10.2011.

Prvi kolokvijum iz Operativnih sistema 2

Kandidat: _____

Broj indeksa: _____ *E-mail:*_____

Kolokvijum traje 2 sata. Dozvoljeno je korišćenje literature.

Zadatak 1	_____/10	*Zadatak 3*	_____/10
Zadatak 2	_____/10	*Zadatak 4*	_____/10

Ukupno: _____/40 = _____%

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 koristi se *Multilevel Feedback-Queue Scheduling* (MFQS), uz dodatni mehanizam koji ga približava ponašanju SJF algoritma. Svakom procesu pridružena je procena trajanja narednog naleta izvršavanja τ koja se izračunava eksponencijalnim usrednjavanjem sa α=1/2, uz odsecanje na ceo broj.

- Postoje tri reda spremnih procesa: HP (*High Priority*), MP (*Medium Priority*) i LP (*Low Priority*).
- Globalni algoritam raspoređivanja je po prioritetu, s tim da HP ima najviši, a LP najniži prioritet.
- Raspoređivanje po redovima je sledeće: za HP je *Round-Robin* (RR) sa vremenskim kvantumom 5, za MP je RR sa vremenskim kvantumom 10, a za LP je FCFS.
- Novoaktivirani proces (deblokiran ili kreiran) smešta se u red prema proceni τ: ako je $\tau \leq 5$, smešta se u HP, ako je $5 < \tau \leq 10$, smešta se u MP, inače se smešta u LP.
- Proces kome je istekao vremenski kvantum premešta se u red nižeg prioriteta od onog iz koga je uzet na izvršavanje. Procena τ se ažurira kada se završi ceo nalet izvršavanja, odnosno kada se proces blokira.

Posmatra se novoaktivirani proces P sa inicijalnom procenom τ=8 koji ima sledeću karakteteristiku narednog izvršavanja (R*n* označava jedan nalet izvršavanja u trajanju *n* jedinica vremena, B označava I/O operaciju):
R2, B, R7, B, R15, B, R1, B, R4
Napisati sekvencu koja označava redove spremnih procesa u kojima redom boravi P, tako da za svako smeštanje procesa P u neki red u sekvenci postoji jedan element. Na primer, odgovor može da bude u obliku: HP, MP, LP, HP, ...

Odgovor: _____

2. (10 poena) Međuprocesna komunikacija pomoću deljene promenljive

Jedna varijanta uslovne sinhronizacije unutar monitora je sledeća. Svaki monitor ima samo jednu, implicitno definisanu i anonimnu (bez imena) uslovnu promenljivu, tako da se u monitoru mogu pozivati sledeće dve sinhronizacione operacije:

- `wait()`: bezuslovno blokira pozivajući proces i oslobađa ulaz u monitor;
- `notify()`: deblokira jedan proces koji čeka na uslovnoj promenljivoj, ako takvog ima.

Projektuje se konkurentni klijent-server sistem. Server treba modelovati monitorom sa opisanom uslovnom promenljivom. Klijenti su procesi koji ciklično obavljaju svoje aktivnosti. Pre nego što u jednom ciklusu neki klijent započne svoju aktivnost, dužan je da od servera traži dozvolu u obliku "žetona" (*token*). Kada dobije žeton, klijent započinje aktivnost. Po završetku aktivnosti, klijent vraća žeton serveru. Server vodi računa da u jednom trenutku ne može biti izdato više od N žetona: ukoliko klijent traži žeton, a ne može da ga dobije jer je već izdato N žetona, klijent se blokira. Napisati kod monitora i procesa-klijenta.

3. (10 poena) Međuprocesna komunikacija razmenom poruka

Na jeziku Java implementirati serverski proces koji predstavlja agenta na aukciji. Ovaj proces treba da „osluškuje“ port 1025 preko koga prima poruku za otvaranje nadmetanja sa početnom cenom. Zatim, nakon zatvaranja prethodne konekcije, po istom portu počinje da prihvata ponude od ostalih učesnika u nadmetanju, pri čemu pamti trenutno najveću ponudu. U svakoj ponudi učesnik se identifikuje svojom vrednošću priključnice (IP adresa i port računara preko koga prima odgovor). Svaka nova ponuda mora biti veća od prethodne, inače se ponuđaču odmah vraća informacija o odbijanju ponude. U suprotnom se vraća poruka da je ponuda prihvaćena i da se čeka krajnji ishod nadmetanja. U slučaju da u međuvremenu pristigne ponuda sa većom vrednošću, vraća se informacija o odbijanju ponude, a ukoliko među prethodnih 5 ponuda ne stigne ni jedna veća, nadmetanje se zatvara, a procesu koji predstavlja učesnika sa najvišom ponudom šalje se poruka o pobedi. Za sinhronizaciju i komunikaciju koristiti priključnice (*sockets*) i mehanizam prosleđivanja poruka (*message passing*).

Rešenje:

4. (10 poena) Upravljanje deljenim resursima

U nekom specijalizovanom sistemu proces može biti „poništen“ (*rolled back*) – ugašen uz poništavanje svih njegovih efekata, i potom pokrenut ispočetka. U ovom sistemu primenjuje se sledeći algoritam sprečavanja mrtve blokade (*deadlock prevention*). Svakom procesu se, prilikom kreiranja, dodeljuje jedinstven identifikator (ID) tako da se identifikatori dodeljuju procesima po rastućem redosledu vremena kreiranja: kasnije kreirani proces ima veći ID. Kada proces P_i sa identifikatorom i zatraži resurs koga drži proces P_j sa identifikatorom j, onda se postupa na sledeći način:

- ako je $i > j$, onda se P_i blokira i čeka da resurs bude oslobođen;
- ako je $i < j$, onda se P_j poništava i pokreće ponovo.

a)(5) Dokazati da se ovim algoritmom sprečava mrtva blokada.

b)(5) Koji ID treba dodeliti poništenom procesu P_j kada se on ponovo pokrene, da bi ovaj algoritam sprečio izgladnjivanje (*starvation*)? Obrazložiti.

Odgovor: