Međuispit iz predmeta *Operacijski sustavi*, 15. 12. 2014.

Ime i prezime

1. (3) Jedna od osnovnih funkcija operacijskog sustava jest omogućavanje višeprogramskog rada koja se ostvaruje korištenjem višedretvenosti. Kako se u takvom sustavu jedna dretva zamjenjuje drugom? Opisati potrebne operacije.

2. (3) Opišite postupak prihvata prekida (što radi sam procesor, što treba programski napravite).

3. (3) U sustavu sa sklopom za prihvat prekida pojavljuju se prekidi: P_1 svakih 10 ms (prvi puta u 3. ms), P_2 svakih 15 ms (prvi puta u 4. ms) te P_3 svakih 30 ms (prvi puta u 6. ms). Ukoliko obrade prekida traju po 3 ms, prihvat prekida (pohrana konteksta te određivanje izvora prekida) 0,5 ms, povratak iz prekida (obnova konteksta) 0,5 ms, prikazati stanje procesora u intervalu [0 ms, 30 ms]. P_3 ima najveći prioritet, slijedi P_2 dok P_1 ima najmanji prioritet.

4. (2) Opisati Lamportov algoritam međusobnog isključivanja (osnovno načelo rada, potrebna struktura podataka).

5. (2) Čemu služi semafor? Koje su osnovne operacije jezgre nad semaforima? Koja je struktura podataka potreba jezgri za ostvarenje semafora?

6. (3) Stanje sustava u nekom trenutku je sljedeće: aktivna dretva je D1, pripravne su D2, D3 i D4, u redu semafora OSEM[1] su D5 i D6. Svi su redovi složeni prema redu prispijeća. Ako tada dretva D1 pozove Odgodi(10) prikazati stanje sustava nakon tog poziva.

7. (2) Sinkronizorati tri dretve: A, B i C tako da u svom radu svoj posao (`posao_dretve_X()`) najprije obave dretve A i B (proizvoljnim redoslijedom ili čak i paralelno – ne ograničiti redoslijed), te tek potom dretva C. Svaka dretva svoj posao obavlja samo jednom.

```
dretva A ()                dretva B ()                dretva C ()
{                          {                          {

   posao_dretve_A();          posao_dretve_B();          posao_dretve_C();

}                          }                          }
```

8. (2) U nekoj pizzeriji je ugrađen automatski sustav upravljanja s peći. Pripremljena sirova jela stavljaju se na pokretnu traku koja vodi do peći. "Peć" uzima jelo i stavlja ga u peć, ako ima mjesta u njoj. Za svako jelo peć pamti potrebno vrijeme pečenja te ga po isteku vadi iz pećnice. Obična pizza zauzima 4 jedinice površine, velika pizza 8 a lazanje 2, dok je ukupni kapacitet peći N mjesta. Pretpostaviti da se jela u peći uvijek mogu presložiti tako da nema fragmentirana prostora. Npr. ako u peći ima slobodno 4 jedinice, tamo će stati još jedna pizza. Simulirati postupak dretvama, gdje svaka dretva predstavlja jedno jelo, tj. kod dretve jest:

```
dretva jelo (tip) //tip = 2 za lazanje, 4 za pizzu te 8 za veliku pizzu
{
    čekaj_ulazak_u_peć (tip);

    pečenje;

    izađi_iz_peći (tip);
}
```

Zamijeniti pozive čekaj* i izađi* potrebnim dijelovima koda. Za sinkronizaciju koristiti semafore i po potrebi zajedničke varijable (1 binarni i 1 opći semafor su dovoljni).

```
dretva čekaj_ulazak_u_peć (tip)          dretva izađi_iz_peći (tip)
{                                        {
```