

1. (2) **Koja je uloga operacijskog sustava u računalnom sustavu (zbog čega se on koristi)?**

višezađačnost -- povećava učinkovitost
skriva složenost sustava -- olakšava korištenje kroz GUI i API

2. (2) **Za ostvarenje višedretvenosti, operacijski sustav mora prekinuti jednu dretvu, "napraviti neke operacije" i onda nastaviti drugu dretvu. Što spada u "napraviti neke operacije"?**

spremiti kontekst prekinute dretve
odabrati drugu dretvu
obnoviti kontekst druge dretve

3. (2) **Koja je prednost korištenja mehanizma prekida za posluživanje ulazno-izlaznih naprava (naspram "radnog čekanja")?**

dok nema zahtjeva od/prema UI napravama,
procesor može raditi nešto drugo (korisno)
"čim" se pojavi zahtjev neke naprave,
ona se mehanizmom prekida može "odmah" poslužiti

4. (2) **Kada je u višedretvenom programu neophodno koristiti međusobno isključivanje?**

kada dretve koriste zajedničke podatke
ne smije se dopustiti paralelna promjena takvih podataka

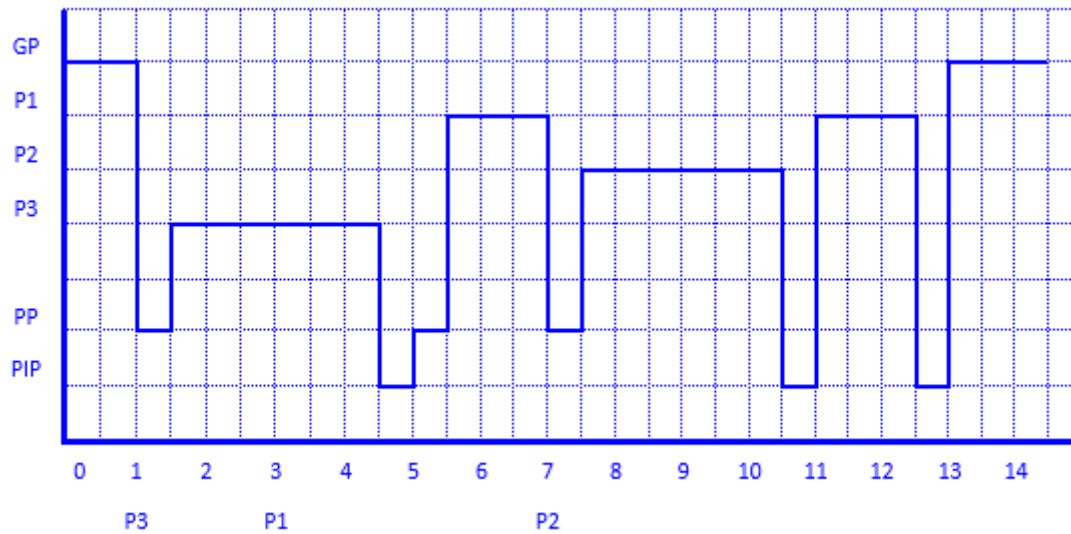
5. (2) **Koje operacije/funkcionalnosti obavlja jezgra operacijskog sustava u jednostavnom modelu jezgre prikazanom na predavanju (za svaku od njih smo definirali dvije jezgrine funkcije)?**

- upravljanje ulazno-izlaznim napravama
- odgoda izvođenja dretvi
- sinkronizacija binarnim i općim semaforima
- (implicitno) raspoređivanje dretvi prema redu prispijeća i/ili prioritetu

6. (2) **Navesti vrijednosti u registru PC nakon izvođenja instrukcija na adresama 1, 3, 7, 150.**

```
1: MOV R0, $1                PC = 2
2: MOV R1, $1
3: CALL $100                 PC = 100
4: ADD R1, R1, R0
5: SUB R0, R1, R0
6: CMP R1, 100
7: BLT $3 ; ako je R1 < 100 skoči na 3    PC = 3
...
100: (ispis sadržaja u R0)
...
150: RET                     PC = 4
```

7. (4) U sustavu koji IMA sklop za prihvat prekida, pojavljuju se zahtjevi: P1 u 3. ms, P2 u 7. ms te P3 u 1. ms. Indeks zadatka označava prioritet (P3 ima najveći). Ukoliko prihvat prekida (PP) traje 0.5 ms, povratak iz prekida (PIP) traje 0.5 ms, a obrade prekida traju po 3 ms, pokazati rad procesora u glavnom programu (GP), obradama prekida (Pi) i procedurama za prihvat prekida (PP) i povratak iz prekida (PIP). Navesti stanje registara sklopa za prihvat prekida u t=9. ms.



Stanje registara sklopa u t=9. ms: KZ=000 TP=011

8. (4) Zadatak Z1 koristi objekte A1, A2, A3, ali promjene radi samo u A1. Zadatak Z2 koristi iste objekte ali radi promjene samo u A2. Zadatak Z3 koristi objekte A1 i A4, s time da radi promjene samo u A4. Ukoliko se zadaci izvode redom Z1→Z2→Z3 rezultat je ispravan. Sinkronizirati zadatke binarnim semaforima tako da se može iskoristiti eventualni paralelizam u izvođenju nezavisnih zadataka, tj. proširiti tekstove zadataka ($T_i \Rightarrow T'_i$) pozivima ČekajBSEM(x) i PostaviBSEM(y). Navesti početne vrijednosti svih korištenih semafora.

	Z1	Z2	Z3	
A1	K	D	D	Z1 mora prvi; Z2 i Z3 nakon njega, paralelno
A2	D	K		
A3	D	D		
A4			K	

T1': T1; PostaviBSEM(1); PostaviBSEM(2)

T2': ČekajBSEM(1); T2

T3': ČekajBSEM(2); T3

početne vrijednosti semafora su 0

(mogli su se jednako koristiti opći semafori)

9. (5) U nekom sustavu redovi pripravnih dretvi, kao i redovi semafora su uređeni prema prioritetu. Prioritet dretve određen je njenim indeksom, gdje veći broj označava veći prioritet. Početno stanje sustava prikazano je u tablici. Pokazati promjene za pozive redom (jedan nakon drugog):
 a) Odogodi (5), b) PrekidUI (1), c) ČekajBSEM(1), d) PostaviOSEM(1).

Red	početno	Odogodi (5)	PrekidUI (1)	ČekajBSEM(1)	PostaviOSEM(1)
Aktivna_D:	4	3	6	6	7
Pripravne_D:	3 2	2	3 2	3 2	6 3 2
Odgođene_D:	1 ² , 5 ¹	1 ² , 5 ¹ , 4 ²	1 ² , 5 ¹ , 4 ²	1 ² , 5 ¹ , 4 ²	1 ² , 5 ¹ , 4 ²
UI[1]:	6	6	–	–	–
OSEM[1]:	7 (.v=0)	7 (.v=0)	7 (.v=0)	7 (.v=0)	– (.v=0)
BSEM[1]:	– (.v=1)	– (.v=1)	– (.v=1)	– (.v=0)	– (.v=0)

10. (5) Dretva *Zaprima* čeka na novi zahtjev klijenta. Kad se zahtjev pojavi potrebno ga je obraditi. To se radi u nekoliko koraka. U prvom se paralelno provjerava klijent i njegove pohranjene postavke (dretva *DohvatiPostavke*) te dohvaća traženi dokument (dretva *DohvatiDokument*). U drugom koraku (zadnjem u ovom pojednostavljenju), koji se može pokrenuti tek nakon prvog, se prilagođava traženi dokument prema postavkama korisnika (dretva *ObradiDokument*) te mu se vraća rezultat. Pseudokod tih dretvi, bez sinkronizacije, prikazan je u nastavku.

Strukture podataka Z1, P2, D2, D3 i P3 su zajedničke za dretve. Proširiti kod sinkronizacijom semaforima. Sinkronizaciju obaviti tako da dretve mogu paralelno raditi svoje operacije – masno označene funkcije (nad istim/različitim zahtjevima). Navesti početne vrijednosti semafora.

```
dretva Zaprima {
  ponavljaaj {
    z = čekaj_zah_tjev()
    ČekajOSEM(1)
    ČekajOSEM(1)
    Z1 = z
    PostaviOSEM(2)
    PostaviOSEM(3)
  }
}

dretva DohvatiPostavke {
  ponavljaaj {
    ČekajOSEM(2)
    ČekajBSEM(1)
    k = Z1
    PostaviBSEM(1)
    PostaviOSEM(1)
    p = dohvati_postavke(k)
    ČekajOSEM(4)
    P2 = p
    PostaviOSEM(6)
  }
}

dretva DohvatiDokument {
  ponavljaaj {
    ČekajOSEM(3)
    ČekajBSEM(1)
    z = Z1
    PostaviBSEM(1)
    PostaviOSEM(1)
    d = dohvati_dokument(z)
    ČekajOSEM(5)
    D2 = d
    PostaviOSEM(6)
  }
}

dretva ObradiDokument {
  ponavljaaj {
    ČekajOSEM(6)
    ČekajOSEM(6)
    p = P2
    d = D2
    PostaviOSEM(4)
    PostaviOSEM(5)
    obradi_pa_vrati(p, d)
  }
}
```

Početne vrijednosti:

```
OSEM[1].v = 2
OSEM[2].v = OSEM[3].v = OSEM[6].v = 0
BSEM[1].v = 1
OSEM[4].v = OSEM[5].v = 1
```