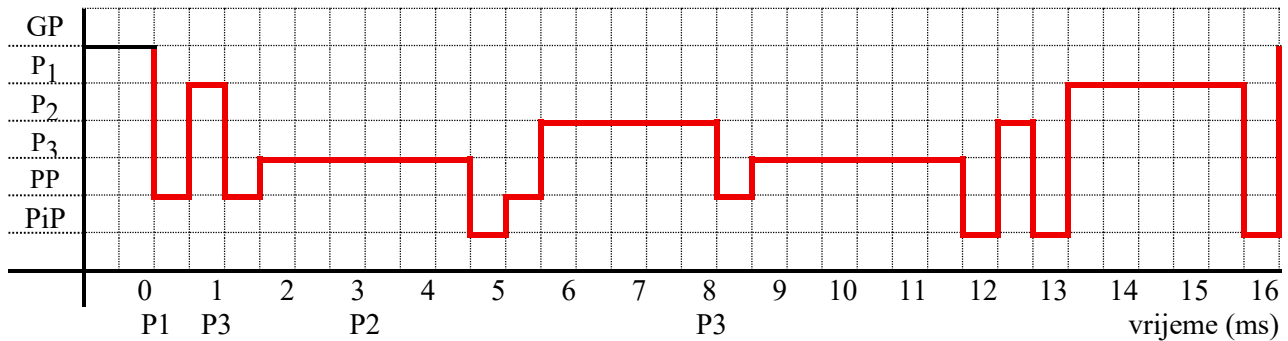


ZEMRIS, 20. 6. 2025.	Ime i prezime	JMBAG
Operacijski sustavi		

Napomena: Sve zadatke rješavati isključivo na ovima listovima. Pisati čitko! Nečitki odgovori su krivi.

1. (6) U nekom sustavu sa sklopom za prihvat prekida javljaju se prekidi: P1 u trenutku $t=0$ ms, P3 u $t=1$ ms,, P2 u $t=3$ ms, te ponovno prekid P3 u $t=8$ ms. Prioritet prekida određen je brojem (P3 ima najveći prioritet). Grafički prikazati aktivnosti procesora u glavnom programu (GP), procedurama za obradu prekida (P_i) koje traju 3 ms te procedurama za prihvat prekida (PP) koje traju 0,5 ms i povratak iz prekida (PiP) koje traju 0,5 ms.



Koja je vrijednost u registru kopija zastavica (KZ) u trenutku $t = 4$ ms?

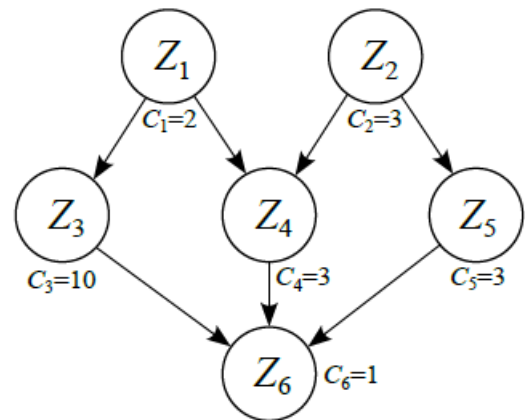
KZ=010

2. (2) Neka naprava koristi sklop za izravni pristup spremniku (DMA). Prosječno u sekundi prenosi 10000 podataka (svaki preko zasebnog sabirničkog ciklusa). Koji postotak sabirničkih ciklusa treba navedena naprava za prijenos tih podataka ako sabirnica radi na 100 MHz?

$$10000/(100 \text{ MHz}) * 100\% = 0,01 \%$$

3. Zadan je sustav zadataka prema slici desno. Uz svaki zadatak zadano je i koliko procesorskog vremena treba za izvođenje (C_n). Sustav treba sinkronizirati binarnim semaforima (BSEM).

- a) (6) Navesti napisati pseudokod za zadatke (T_i') uz pretpostavku da je koristan posao zadatka Z_i označen s T_i .
b) (2) Koliko će trajati izvođenje sustava na četvero-procesorskom sustavu (uz ostvarenu sinkronizaciju)?



a)

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

T2': T2; PostaviBSEM(3), PostaviBSEM(4)

T3': ČekajBSEM(1); T3; PostaviBSEM(5)

T4': ČekajBSEM(2); ČekajBSEM(3); T4; PostaviBSEM(6)

T5': ČekajBSEM(4); T5; PostaviBSEM(7)

T6': ČekajBSEM(5); ČekajBSEM(6); ČekajBSEM(7);

b)

najduža grana je $Z1 \rightarrow Z3 \rightarrow Z6 = 2+10+1 = 13$ jedinica vremena

4. (8) U promatranom trenutku stanje sustava je opisano u stupcu „Početno stanje“. Redovi pripravnih i semafora organizirani su po prioritetu gdje veći broj dretve označava veći prioritet. Pozivi koji se tada događaju su navedeni u tablici. Pokazati stanje sustava nakon svakog od ovih poziva (svaki idući se odnosi na prethodni stupac, na stanje nakon prethodne jezgrine funkcije).

	Početno stanje	Nakon Odgodi(3)	Nakon ZapočniUI(1)	Nakon PostaviOSEM(S)	Nakon Prekid_sata()
Aktivna_D	7	5	3	4	4
Pripravne_D	5 3 1	3 1	1	3 1	3 1
Odgodene_D	6 ⁵	7 ³ 6 ²	7 ³ 6 ²	7 ³ 6 ²	7 ² 6 ²
OSEM[S]	4 (.v=0)	4 (.v=0)	4 (.v=0)	- (.v=0)	- (.v=0)
Red UI[1]	-	-	5	5	5

5. (4) U nekom sustavu se nalaze dva proizvođača i tri potrošača koji komuniciraju preko zajedničkog međuspremnik. Kod proizvođača zadan je lijevo. Navesti kod potrošača desno.

```

proizvođač{
    ponavljaaj {
        P = stvori poruku()
        Čekaj_OSEM(2)
        Čekaj_BSEM(1)
        M[ULAZ] = P
        ULAZ = (ULAZ + 1) MOD N
        Postavi_BSEM(1)
        Postavi_OSEM(1)
    }
    do zauvijek
}

potrošač {
    ponavljaaj {
        ČekajOSEM(1)
        ČekajBSEM(2)
        P = M[IZLAZ]
        IZLAZ = (IZLAZ + 1) MOD N
        PostaviBSEM(2)
        PostaviOSEM(2)
        obradi poruku(P)
    }
    do zauvijek
}

```

6. (6) U nekom sustavu se izvode dretve 1a, 1b, 2a, 2b, 3a, 3b. Sve se raspoređuju prema prioritetu kao primarnom kriteriju te podjelom vremena kao sekundarnom uz kvant Tq=1 (način SCHED RR). Prioritet dretvi određen je njenim indeksom - 3a i 3b imaju najveći prioritet 3. Dretve se javljaju u sustavu u trenutcima: (t, D) = {(0, 2a), (1, 1a), (2, 2b), (3, 3a), (5, 1b), (9, 3b)}. Pokazati izvođenje dretvi do t=20 ukoliko svaka dretva treba četiri jedinice vremena za svoje izvođenje. U trenutcima kad se više događaja događa istovremeno proizvoljno odabrati redoslijed.

Pripravne dretve (nije potrebno pisati)	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
						1b	1b														
				1a	1a	1a	1a	1b	1b	1b	1b	1b	1b								
			1a	2a	2a	2a	2a	1a	1a	1a	1a	1a	1a	1b	1b	1b					
		1a	2b	2b	2b	2b	2b	2a	2b	2b	2b	2b	2b	1a	1a	1a	1b	1a	1b	1a	
Aktivna	2a	2a	2a	3a	3a	3a	3a	2b	2a	3b	3b	3b	3b	2b	2b	2b	1a	1b	1a	1b	
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
	2a	1a	2b	3a		1b				3b											

7. (4) Za neki proces zadana je tablica prevođenja (desno). Ako je veličina stranice 4096 B koju će fizičku adresu generirati sklop za logičke adrese 0x00281, 0x02810 te 0x04333?

0x00281 => 0x77281
 0x02810 => 0x88810
 0x04333 => promašaj = prekid

str.	okvir	bit pris.
0	0x77	1
1	-	0
2	0x88	1
3	0x99	1
4	-	0

8. (8) U sustavu koji koristi upravljanje spremnikom stranicenjem s 10 MB spremnika javljaju se zahtjevi za pokretanjem procesa te završetka prema redu: +P3, +P7, -P3, +P4, -P7, +P3, +P5, -P4, -P3, +P8. Plus ispred oznake procesa +PX označava zahtjev za pokretanje procesa PX koji treba X MB radnog spremnika, dok -PX označava kraj izvođenja tog procesa. Veličina stranice je 1 MB. OS neće ni jednom procesu dati više od pet okvira, ali niti manje od tri, tj. ako nema barem tri prazna, pokretanje procesa će se odgoditi dok se stanje ne promijeni. OS ne oduzima dodijeljene okvire procesima, ali će ih još dati onima koji nisu dobili četiri/pet (a mogu koristiti četiri/pet) kad se pojave prazni. Prikazati stanje spremnika nakon svakog od zahtjeva.

zahtjevi										
+P3	3	3	3							
+P7	3	3	3	7	7	7	7	7		
-P3				7	7	7	7	7		
+P4	4	4	4	7	7	7	7	7	4	
-P7	4	4	4						4	
+P3	4	4	4	3	3	3			4	
+P5	4	4	4	3	3	3	5	5	4	5
-P4	5	5		3	3	3	5	5		5
-P3	5	5					5	5		5
+P8	5	5	8	8	8	8	5	5	8	5

9. Neka datoteka pohranjena je na disku koji koristi NTFS. U opisniku te datoteke nalazi se tablica desno.

VCN	LCN	#
0	4000	3000
3000	40000	5000
8000	50000	2000

a) (2) Ako je blok velik 16 KB, koliko je velika datoteka?

$$(3000+5000+2000)*16 \text{ KB} = 160\,000\,000 \text{ KB} = 156\,250 \text{ MB} = 154,6 \text{ GB}$$

b) (2) U kojem bloku diska se nalazi 150 000 000. bajt datoteke?

$$150\,000\,000 \text{ B} / 16 \text{ KB} = 150\,000\,000 / (16*1024) = 9155,27 \Rightarrow \text{u bloku datoteke 9156}$$

blok 9156 je opisan u trećem retku te je blok diska: 50000 + (9156-8000) = 51156

10. (2) Neki sustav koristi RAID5 s 10 diskova, a drugi RAID5 s 20. Koji od njih je pouzdaniji ako koriste diskove istih svojstava? Obrazložiti.

pouzdaniji je onaj s manje diskova, RAID5 s 10 diskova

više diskova => veća vjerojatnost kvara

11. (2) Na raspolaganju je 20 diskova kapaciteta 6 TB svaki. Zahtjevi nekog sustava su: kapacitet 50 TB, maksimalna pouzdanost i dugotrajni rad. Koji RAID ćete odabrati i s koliko diskova?

za 50 TB treba 9*6 TB korisna kapaciteta

za maksimalnu pouzdanost treba odabrati RAID6

$$9 + 2 = 11 \text{ diskova}$$

12. (8 bodova) Problem sličan problemu pet filozofa pojavljuje se u „Restoranu na kraju svemira“. U restoranu se nalazi samo jedan dugi stol kapaciteta N mjesta za koji gosti mogu sjesti ako ima praznih mjesta (s iste strane stola, s pogledom na „kraj svemira“). Na svakom praznom mjestu već se nalazi prazan tanjur s priborom. Na stolu se nalaze tri zdjele s tri tipa jela – na svakoj zdjeli piše što sadrži. Restoran poslužuje razna jela, ali u jednom danu samo tri različita jela. Svaki gost želi samo jedan tip jela - ako na stolu tog dana nema zdjele s njegovim tipom, gost odlazi (ne sjeda za stol). U protivnom će sjesti. Ako u zdjeli hrane koju on želi ima hrane, onda uzima jednu porciju, jede i nakon toga odlazi. Ako je njegova zdjela prazna, on poziva konobara koji će kad dođe na svaku zdjelu dodati po tri obroka. Kad jedan gost ode, robot poslužitelj dođe, uzme pribor i postavi čisti (tek tad može novi gost na to mjesto). Simulirati goste, kuhara i robota, dretvama `gost(tip)`, `robot()`, `konobar()` i sinkronizirati ih monitorima (može sintaksa s `labosa mutex_lock` i sl. ili s predavanja `Uđi_u_monitor()` i sl.). Rješavati na poledini, tj. nadopuniti pseudokod.

```

tip[3]={a,b,c} //tip hrane na stolu
porcija[3]={5,5,5} //koliko porcija odgovarajućeg tipa ima na stolu
mjesto[N] = {0,0,...0} //je li to mjesto slobodno(0), zauzeto(1), slobodno-ali-prljavo(2)
slobodnih = N //koliko ima slobodnih, radi jednostavnosti "dok je" petlje :)

```

m - monitor

red_stol, red_hrana, red_kuhara, red_robota - redovi uvjeta

```

dretva gost(tip) {
    ako je (tip[0] == tip)
        zdjela = 0
    inače ako je (tip[1] == tip)
        zdjela = 1
    inače ako je (tip[2] == tip)
        zdjela = 2
    inače {
        izadi_iz_funkcije
    }
    mutex_lock(m)
    dok je (slobodnih == 0)
        cond_wait(red_stol, m)
    slobodnih--
    za i=1 do N
        ako je mjesto[i] == 0
            moje_mjesto = i
    mjesto[moje_mjesto] = 1
    sjedni_za_stol
    dok je porcija[zdjela] == 0 {
        cond_signal(red_kuhara)
        cond_wait(red_hrana, m)
    }
    porcija[zdjela]--
    mutex_unlock(m)
    jedi;
    mutex_lock(m)
    mjesto[moje_mjesto] = 2
    cond_signal(red_robota)
    mutex_unlock(m)
}

```

```

dretva konobar() {
    ponavljaaj {
        mutex_lock(m)
        cond_wait(red_kuhara, m)
        porcija[0] += 5
        porcija[1] += 5
        porcija[2] += 5
        cond_broadcast(red_hrana)
        mutex_unlock(m)
    }
}

dretva robot() {
    ponavljaaj {
        mutex_lock(m)
        cond_wait(red_robota, m)
        za i=1 do N {
            ako je (mjesto[i] == 2) {
                počisti-mjesto(i)
                mjesto[i] = 0
                slobodnih++
            }
        }
        cond_broadcast(red_stol)
        mutex_unlock(m)
    }
}

```