VAŽNA NAPOMENA:

- Ova pitanja NISU materijal za pripremu kolokvijuma, već materijal za proveru znanja i vežbanje zadataka.
- Za pripremu kolokvijuma koristite PDF materijale koji su raspoloživi na stranici predmeta (sekcija: predavanja).

9.1. Objasnite zašto SSTF favorizuje cilindre diska koji se nalaze bliže polovini radijusa ploče u odnosu na periferne i centralne cilindre.

Cilindri diska koji se nalaze bliže polovini radijusa ploče imaju najmanju srednju udaljenost od ostalih cilindara. Iz tih razloga, glave za čitanje i pisanje imaju tendenciju udaljavanja od centralnih i perifernih cilindara.

9.2. Posmatrajte disk sa 5000 cilindara označenih brojevima od 0-4999. Glave za čitanje i pisanje se trenutno nalaze cilindru 143, a prethodno su bile pozicionirane na cilindru 125. Trenutni raspored reda za rad sa diskom u FIFO poretku je: 86, 1470, 913, 1774, 948, 1509, 1022, 1750, 130. Odredite redosled opsluživanja disk zahteva i ukupan pomeraj (ukupan broj cilindara obuhvaćen *seek* sekvencom), ukoliko se za raspoređivanje zahteva koriste sledeći algoritmi: (a) FCFS, (b) SSTF, (c) SCAN, (d) LOOK, (e) C-SCAN, (f) C-LOOK. Za početni trenutak usvojite momenat u kome su glave diska na 143 cilindru.

(a) Raspored zahteva prema FCFS algoritmu je (brojevi u zagradi predstavljaju pomeraj koji glave diska prave pri prelasku na sledeći cilindar, izražen u broju cilindara):

143, 86 (57), 1470 (1384), 913 (557), 1774 (861), 948 (826), 1509 (561), 1022 (487), 1750 (728), 130 (1620).

Ukupan pomeraj glava: 7801 cilindara.

(b) Raspored zahteva prema SSTF algoritmu je: 143, 130 (13), 86 (44), 913 (827), 948 (35), 1022 (74), 1470 (448), 1509 (39), 1750 (241), 1774 (24).

Ukupan pomeraj glava: 1745 cilindara.

(c) Raspored zahteva prema SCAN algoritmu je: 143 , 913 (770), 948 (35), 1022 (74), 1470 (448), 1509 (39), 1750 (241), 1774 (24), 4999 (3225), 130 (4869), 86 (44).

Ukupan pomeraj glava: 9769 cilindara.

(d) Raspored zahteva prema LOOK algoritmu je: 143 , 913 (770), 948 (35), 1022 (74), 1470 (448), 1509 (39), 1750 (241), 1774 (24), 130 (486), 86 (44).

Ukupan pomeraj glava: 3319.

(e) Raspored zahteva prema C-SCAN algoritmu je: 143 , 913 (770), 948 (35), 1022 (74), 1470 (448), 1509 (39), 1750 (241), 1774 (24), 4999 (3225), 0 (4999), 86 (86), 130 (44).

Ukupan pomeraj glava: 9813.

(f) Raspored zahteva prema C-LOOK algoritmu je: 143 , 913 (770), 948 (35), 1022 (74), 1470 (448), 1509 (39), 1750 (241), 1774 (24), 86 (1688), 130 (44).

Ukupan pomeraj glava: 3363.

9.3. Data su četiri identična diska kapaciteta 10GB sa sledećim karakteristikama:

- prosečno vreme pozicioniranja (*average seek time*): 6.3msec,
- pozicioniranje s kraja na kraj (*full stroke seek*): 15msec,
- pozicioniranje na sledeći cilindar (*track to track seek*): 0.8msec,
- ugaona brzina: 7200rpm (8.33 msec),
- pristup medijumu (*sustain data rate*): 24.5MB/sec,
- propusna moć: 160MB/sec,
- srednje vreme otkaza (*mean time to failure*): 200.000 časova (23 godine),
- cena: $50.

Od ovih diskova formiran je RAID-0 sa *stripe* jedinicom veličine 4KB (8 blokova).

a. Skicirati raspored stripe jedinicama ovog RAID sistema. Koliki je kapacitet ovog RAID-a? Koliko blokova ima ovaj RAID, a koliko stripe jedinica?

b. Odredite na kom disku se nalazi logički blok 153.

c. U redu za korišćenje diska nalaze se redom zahtevi sa sledećim veličinama transfera: 1K, 8K, 12K, 2K, 4K. Koliko zahteva može opslužiti ovaj RAID istovremeno za ovaj red, a koliko uopšte?

d. Odredite cenu po gigabajtu i stepen iskorišćenja prostora.

(a) Sledeća tabela predstavlja raspored stripe jedinica datog sistema:

disk0	**disk1**	**disk2**	**disk3**
0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15
16	17	18	19
20	21	22	23
...	...	...	...
N-4	N-3	N-2	N-1

Kapacitet ovog RAID-0 sistema je 40GB:

broj blokova = broj diskova · broj blokova na disku,

broj blokova = 4 · (10GB/512) = 78125000 blokova,

broj traka = broj blokova / veličina trake u blokovima,

broj traka = 78125000 / 8 = 9765625,

(b) Prvo se određuje kojoj traci pripada blok.

Stripe = Integer (logički blok / veličina trake) = Integer(153/8) = 19.

Disk = (stripe % broj diskova) = moduo (19/4) = 3 (logički blok se nalazi na disku broj 3, a kako se diskovi numerišu počev od nule, to je četvrti disk u RAID-u).

(c) I/O queue = (1K, 8K, 12K, 2K, 4K)

Teorijski, RAID 0 može opslužiti onoliko zahteva koliko ima diskova u RAID-u, pod uslovom da je transfer manji od veličine trake i da se trake nalaze na različitim diskovima. U ovom slučaju RAID trenutno može da izvršava najviše tri zahteva (1K, 2K, 4K), pod uslovom da se zahtevi odnose na trake smeštene na različitim diskovima. Teorijski, ovaj RAID može izvršiti 4 zahteva simultano.

(d) Cena po GB = ukupna cena / ukupni kapacitet.

Cena po GB = 4 x 50$ / 4 x 10GB = 5$/GB.

Stepen iskorišćenja prostora za RAID 0 je 100%.

9.4. Data su četiri identična diska kapaciteta 10GB sa sledećim karakteristikama:

- prosečno vreme pozicioniranja (*average seek time*): 6.3msec,
- pozicioniranje s kraja na kraj (*full stroke seek*): 15msec,
- pozicioniranje na sledeći cilindar (*track to track seek*): 0.8msec,
- ugaona brzina: 7200rpm (8.33 msec),
- pristup medijumu (*sustain data rate*): 24.5MB/sec,
- propusna moć: 160MB/sec,
- srednje vreme između otkaza: 200.000 časova (23 godine),
- cena: $50.

Od ovih diskova formiran je RAID-4 sa *stripe* jedinicom veličine 2KB.

a. Skicirati ovaj RAID sistem. Koliki je kapacitet, koliko logičkih blokova, a koliko *stripe* jedinica ima ovaj RAID?

b. Odredite na kom se disku se nalazi logički blok 3.945.

c. U redu za korišćenje diska nalaze se redom zahtevi sa sledećim veličinama transfera: 1K, 8K, 12K, 2K, 4K. Koliko zahteva može opslužiti ovaj RAID istovremeno za ovaj red, a koliko uopšte?

d. Odredite cenu po gigabajtu i stepen iskorišćenja prostora..

(a) Sledeća tabela ilustruje dati RAID (trake veličine 2KB):

disk0	disk1	disk2	disk3
0	1	2	P
3	4	5	P
6	7	8	P
9	10	11	P
12	13	14	P
15	16	17	P
...	...	...	...
N-3	N-2	N-1	P

Kapacitet ovog RAID-1 sistema je 30GB:

broj blokova = (broj diskova - 1) · broj blokova na disku,

broj blokova = (4-1) · (10GB/512) = 58593750 blokova,

broj traka = broj blokova / veličina trake u blokovima,

broj traka = 58593750 / 4 = 146.484.375.

(b) Prvo se određuje kojoj traci pripada blok.

Stripe = Integer (logički blok / veličina trake) = Integer(3945/4) = 986.

Disk = (stripe % (broj diskova-1)) = moduo (986/3) = 2 (logički blok se nalazi na disku broj 2, a kako se diskovi numerišu počev od nule to je treći disk u RAID-u).

(c) I/O queue = (1K, 8K, 12K, 2K, 4K).

Teorijski, RAID 4 može opslužiti onoliko zahteva koliko ima diskova u RAID-u, umanjeno za 1, pod uslovom da je transfer manji od veličine trake i da se trake nalaze na različitim diskovima. U ovom slučaju RAID trenutno može da izvršava najviše dva zahteva (1K, 2K), pod uslovom da se zahtevi odnose na trake smeštene na različitim diskovima. Teorijski, ovaj RAID može izvršiti 3 zahteva simultano.

(d) Cena po GB = ukupna cena / ukupni kapacitet.

Cena po GB = 4 x 50$ / 3 x 10GB = 6.66$/GB.

Stepen iskorišćenja prostora za RAID 4 je 75%.

9.5. Data su četiri identična diska kapaciteta 10GB, sa sledećim karakteristikama:

- prosečno vreme pozicioniranja (*average seek time*): 6.3msec,
- pozicioniranje s kraja na kraj (*full stroke seek*): 15msec,
- pozicioniranje na sledeći cilindar (*track to track seek*): 0.8msec,
- ugaona brzina: 7200rpm (8.33 msec),
- pristup medijumu (*sustain data rate*): 24.5MB/sec,
- propusna moć: 160MB/sec,
- srednje vreme između otkaza: 200.000 sati (23 godine),
- cena: $50.

Od navedenih diskova formiran je RAID-5 sa *stripe* jedinicom veličine 4KB (8 blokova).

a. Skicirati ovaj RAID sistem. Koliki je kapacitet, koliko logičkih blokova, a koliko *stripe* jedinica ima ovaj RAID?

b. Odredite na kom disku se nalazi logički blok 111.

c. U redu za korišćenje diska nalaze se redom zahtevi koji imaju sledeće veličine transfera: 1K, 8K, 12K, 2K, 4K. Koliko zahteva može opslužiti ovaj RAID istovremeno za ovaj red, a koliko uopšte?

d. Odredite cenu po gigabajtu i stepen iskorišćenja prostora.

(a) Sledeća tabela ilustruje dati RAID (trake veličine 2KB):

disk0	**disk1**	**disk2**	**disk3**
0	1	2	P
3	4	P	5
6	P	7	8
P	9	10	11
12	13	14	P
15	16	P	17
...	...	...	...
N-3	N-2	N-1	P

Kapacitet ovog RAID-1 sistema je 30GB:

broj blokova = (broj diskova - 1) · broj blokova na disku,

broj blokova = (4-1) · (10GB/512) = 58.593.750 blokova

broj traka = broj blokova / veličina trake u blokovima,

broj traka = 58.593.750 / 8= 7.324.218.

(b) Prvo se određuje kojoj traci pripada blok.

Stripe = Integer (logički blok / veličina trake) = Integer(111/8) = 13.

Iz prethodne tabele se vidi da se traka br.13 nalazi na disku 1.

(c) I/O queue = (1K, 8K, 12K, 2K, 4K).

Teorijski, RAID 5 može opslužiti onoliko zahteva koliko ima diskova u RAID-u, umanjeno za 1, pod uslovom da je transfer manji od veličine trake i da se trake nalaze na različitim diskovima. U ovom slučaju RAID trenutno može da izvršavaju najviše tri zahteva (1K, 2K, 4K), pod uslovom da se zahtevi odnose na trake smeštene na različitim diskovima. Teorijski, ovaj RAID može izvršiti 3 zahteva simultano.

(d) Cena po GB = ukupna cena / ukupni kapacitet.

Cena po GB = 4 x 50$ / 3 x 10GB = 6.66$/GB.

Stepen iskorišćenja prostora za RAID 5 je 75%.

9.6. U čemu su prednosti i mane realizacije *swap* prostora (a) u formi posebne particije, (b) u formi datoteke u postojećem sistemu datoteka ?

(a) Prednosti: upravljanje *swap* prostorom zaobilazi rutine za rad sa datotekama, tako da se *swap* prostoru brže pristupa.

Mane: zahteva particionisanje diskova.

(b) Prednosti: ne zahteva particionisanje diskova.

Mane: upravljanje *swap* prostorom obavlja se preko rutina za rad sa datotekama što unosi kašnjenje.

10.1. U čemu je razlika između mrežnih i distribuiranih operativnih sistemam?

Pod mrežnim operativnim sistemima, korisnik može pristupiti udaljenim resursima na dva načina: procedurom prijavljivanja na odgovarajući udaljeni računar i transferom datoteka sa udaljene mašine na sopstvenu. U distribuiranim operativnim sistemima korisnik pristupa udaljenim resursima kao da su lokalni.

10.2. Koje su tri vrste migracija karakteristične za distribuirane operativne sisteme ?

- migracija podataka - korisnik koji želi da pristupi udaljenoj datoteci dobija: (1) kopiju cele datoteke ili (2) zahtevani deo datoteke,
- migracija izračunavanja - proces može inicirati obradu na udaljenoj strani (pomoću rpc poziva ili slanjem poruka), a zatim pokupiti rezultate,
- migracija procesa - proces kreiran na lokalnoj strani ne mora se izvršavati strogo na svom računaru, već se ceo proces ili neki njegovi delovi mogu izvršavati na drugim računarima.

10.3. a. Definišite relaciju *happened before* (relaciju označite simbolom ->).

b. Predstavite relacijom događaje slanja poruka..

c. Neka je svakom fizičkom događaju dodeljena vremenska oznaka - TS (timestamp). Kako se relacija *happened before* može definisati pomoću ovih oznaka?

(a) Ako su A i B događaji istog procesa i ako je događaj A izvršen pre događaja B, tada važi A -> B.

(b) Ako je događaj A slanje poruke, a događaj B primanje te poruke, pri čemu različiti procesi šalju i primaju poruku, tada je A -> B.

(c) Relacija hapended before se za svaki par događaja A i B definiše na sledeći način: ako je A->B, tada je TS događaja A manji od TS događaja B, tj. TS(A)<TS(B).

10.4. Razmotriti sledeći slučaj konkurentnih atomskih transakcija, koje implemntiraju TS protokol: dat je zapis Q u baznoj datoteci koji ima sledeće *read* i *write* vremenske oznake:

- timestamp zadnjeg čitanja: R(Q)=100,

- timestamp zadnjeg upisa: W(Q)=150.

Na izvršenje čeka transakcija čitanja Ti, sa vremenskom oznakom TS(Ti)=50.

a. Iz koje strukture će se dogoditi čitanje: iz Q zapisa ili iz log datoteke (dnevnika)?

b. Koje su nove vrednosti R(Q) i W(Q)?

(a) TS(Ti)=50, W(Q)=150 $\Rightarrow$ TS(Ti) < W(Q)

Transakcija Ti traži vrednost Q koja pripada prošlosti, i kao takva je prepisana. Čitanje se odbacuje, a vrednost se dobija primenom *roll-back* metode, za Ti - čitanje se obavlja iz dnevnika transakcija pomoću roll-back metode.

(b) Ti nije imala uticaj na zapis Q, timestamp zadnjeg čitanja i upisa za zapis Q ostaju netaknuti: R(Q)=100, W(Q)=150

10.5. Razmotriti sledeći slučaj konkurentnih atomskih transakcija, koje implemntiraju TS protokol: dat je zapis Q u baznoj datoteci koji ima sledeće *read* i *write* vremenske oznake:

- timestamp zadnjeg čitanja: R(Q)=100,
- timestamp zadnjeg upisa: W(Q)=150.

Na izvršenje čeka transakcija čitanja Ti, sa vremenskom oznakom TS(Ti)=200.

a. Iz koje strukture će se dogoditi čitanje: iz Q zapisa ili iz log datoteke (dnevnika) ?

b. Koje su nove vrednosti R(Q) i W(Q) ?

(a) TS(Ti)=50, W(Q)=150 $\Rightarrow$ TS(Ti) ≥ W(Q)

Zahtev je korektan, čitanje se odvija iz zapisa Q, a posle toga se ažurira vremenska oznaka R(Q).

(b) R(Q)=200, W(Q)=150

10.6. Razmotriti sledeći slučaj konkurentnih atomskih transakcija, koje implemntiraju TS protokol: dat je zapis Q u baznoj datoteci koji ima sledeće *read* i *write* vremenske oznake:

- timestamp zadnjeg čitanja: R(Q)=100,
- timestamp zadnjeg upisa: W(Q)=150.

Na izvršenje čeka transakcija upisa Ti, sa vremenskom oznakom TS(Ti)=50.

a. Da li će doći do upisa i gde?

b. Koje su nove vrednosti R(Q) i W(Q)?

(a) TS(Ti)=50, R(Q)=100 $\Rightarrow$ TS(Ti) < R(Q)

Zahtev je nekorektan, jer transakcija pokušava upis u nešto što je već trebalo da bude pročitano. Upis u Q se odbacuje, Ti obavlja upis u dnevnik transakcija, a sve transakcije koje su čitale pogrešno (sve transakcije čitanja između TS(Ti) i sledećeg upisa u Q) moraju da se nateraju na *roll-back*.

(b) Ti nije imala uticaj na zapis Q, timestamp zadnjeg čitanja i upisa za zapis Q ostaju netaknuti: R(Q)=100, W(Q)=150

10.7. Razmotriti sledeći slučaj konkurentnih atomskih transakcija, koje implemntiraju TS protokol: dat je zapis Q u baznoj datoteci koji ima sledeće *read* i *write* vremenske oznake:

- timestamp zadnjeg čitanja: R(Q)=100,
- timestamp zadnjeg upisa: W(Q)=150.

Na izvršenje čeka transakcija upisa Ti, sa vremenskom oznakom TS(Ti)=120.

a. Da li će doći do upisa i gde?

b. Koje su nove vrednosti R(Q) i W(Q)?

(a) TS(Ti)=120, R(Q)=100 $\Rightarrow$ TS(Ti) > R(Q)

TS(Ti)=120, W(Q)=100 $\Rightarrow$ TS(Ti) < W(Q)

Zahtev je nekorektan jer transakcija Ti pokušava upis zastarele vrednosti. Upis u Q se odbacuje, Ti obavlja upis u dnevnik transakcija, ali ni jedna transakcija čitanja ne mora da radi *roll-back*.

(b) Ti nije imala uticaj na zapis Q, timestamp zadnjeg čitanja i upisa za zapis Q ostaju netaknuti: R(Q)=100, W(Q)=150

10.8. Razmotriti sledeći slučaj konkurentnih atomskih transakcija, koje implemntiraju TS protokol: dat je zapis Q u baznoj datoteci koji ima sledeće read i write vremenske oznake:

- timestamp zadnjeg čitanja: R(Q)=100,
- timestamp zadnjeg upisa: W(Q)=150.

Na izvršenje čeka transakcija upisa Ti, sa vremenskom oznakom TS(Ti)=180.

a. Da li će doći do upisa i gde?

b. Koje su nove vrednosti R(Q) i W(Q)?

(a) TS(Ti)=50, W(Q)=150 $\Rightarrow$ TS(Ti) ≥ W(Q)

Zahtev je korektan, transakcija obavlja upis u Q, a posle toga se ažurira vremenska oznaka W(Q).

(b) R(Q)=200, W(Q)=180

10.9. a. Koje se metode za prevenciju zastoja zasnovane na vremenskim oznakama koriste u distribuiranim sistemima?

b. Koja od ovih metoda koristi pretpražnjenje (oduzimanje resursa)?

(a) 1. Metoda *wait-die* (stariji proces čeka na mlađeg da otpusti resurs). Proces Pi, koji traži resurs dodeljen procesu Pj, čekaće da proces Pj taj resurs otpusti samo ako je TS(Pi)>TS(Pj). U protivnom, proces Pi se podvrgava *rollback* operaciji.

2. Metoda *wound-wait* (stariji proces nikada ne čeka na mlađe). Proces Pi čekaće na resurs dodeljen procesu Pj samo ako je Pi mlađi, odnosno ako je TS(Pi)<TS(Pj). U protivnom, Pj se podvrgava rollback operaciji, a resurs mu se oduzima i predaje procesu Pi.

(b) Metoda *wound-wait*.

10.10. Ako je red prispelih transakcija:

read TS(T1)=150,

read TS(T2)=80,

write TS(T3)=75,

write TS(T4)=160,

odrediti kakav će redosled napraviti timestamp protokol, u cilju izbegavanja *roll-back* efekata.

Da bi se izbegli kaskadne roll-back operacije, u sinhronizaciji transakcija čitanja i upisa moraju se poštovati sledeća pravila:

- transakcija Ti za čitanje podataka x mora biti odložena, ako postoji transakcija Tj koja treba da izvrši upis u x, takva da je TS(Tj)<TS(Ti) (odloži čitanje ako postoji upis pre čitanja),
- transakcija Ti za upis podataka u x mora da biti odložena ako postoji transakcija Tj koja čita ili upisuje podatke u x, takva da je TS(Tj)<TS(Ti) (odloži upis ukoliko pre upisa postoji čitanje ili drugi upis).

Zadatak se rešava jednostavno, prvo ide najstari upis, zatim čitanje i tako redom. Optimalna sekvenca je: T3, T2, T1, T4.

10.11. Ako je red prispelih transakcija:

read TS(T1)=150,

read TS(T2)=60,

write TS(T3)=75,

read TS(T4)=160,

odrediti kakav će redosled napraviti timestamp protokol, u cilju izbegavanja *roll-back* efekata.

Optimalne sekvence su: (1) T2, T3, T1, T4 i (2) T2, T3, T4, T1. Redosled transakcija T1 i T4 je nebitan jer su i T1 i T4 transakcije čitanja.

10.12. Ako je red prispelih transakcija:

read TS(T1)=150,

read TS(T2)=60,

read TS(T3)=75,

read TS(T4)=160,

odrediti kakav će redosled napraviti timestamp protokol, u cilju izbegavanja *roll-back* efekata.

Sve sekvence mogu se smatrati optimalnim jer nema transakcija upisa.