Skripta iz ArOs-a

Napomena: Ova skripta pokriva materijale sa vezbi i predavanja, kao i uradjene primere iz knjige (pitanja na kraju svake glave). Skoro sva pitanja su uradjena, jedino je izostavljena glava br. 12 (poslednja glava) jer nismo stigli da to uradimo. Izvinjavamo se unapred zbog gresaka u skripti (uglavnom su to greske u kucanju), trudili smo se da greske u zadacima maksimalno ispravimo. Ovaj deo skripte obuhvata oblast Operativnih Sistema (ukoliko neko hoce da sprema ceo ispit, potrebna ce mu biti i prva skripta by Oziris). Na prvim stranama knjige dat je pregled pitanja i zadataka (sa resenjima) datih na vezbama i predavanjima (uglavnom se ta pitanja preklapaju), sa datumima. Zelim da se zahvalim svim divnim ljudima koji su radili na skripti, (Smiljana, Zeljka, Branka, Vanja) i jos jednom pokazali meni i drugima koliko kolegijalnost znaci.

Ukoliko ste uocili greske u skripti, nesto vam nije jasno obratite mi se na mail, rado cu vam pomoci.

mail: oziris@lolbox.org

Oziris

1) Nacini rada OS-a?

   - batch (paketna obrada);
   - time-sharing;
   - real-time obrada.

2) Date su dinamicke memorije sledecih velicina:
   100 KB
   500 KB
   200 KB
   300 KB
   600 KB
   sortirane po pocetnim memorijskim adresama u rastucem redosledu. Kako ce operativni sistem da podeli ove particije sledecim procesima, cije su zahtevane velicine memorije
   212 KB
   417 KB
   112 KB
   426 KB
   ako su procesi navedeni po redosledu dolaska i ako se koristi algoritam najboljeg uklapanja, a potom i algoritam prvog uklapanja.

   a) best fit
      212 KB <- P1 212 -> 300
      417 KB <- P2 417 -> 500
      112 KB <- P3 112 -> 200
      426 KB <- P4 426 -> 600

   b) first fit (prvo uklapanje)
      212 -> 500 KB
      417 -> 600 KB
      112 -> 200 KB
      426 -> /

3) Dat je sledeci trag adrese
   0 7 2 1 7 3 2 4 7 2 5 1
   Pokazite koliko ce stranicnih prekida biti napravljeno u slucaju da program na raspolaganju ima tri okvira, a zatim cetiri okvira, i da li je pritom bolje koristiti FIFO ili LRU algoritam?

   Napomena:
   FIFO – izbacuje stranicu koja je prva koriscena
   LRU - izbacuje stranicu koja najduze nije koriscena
   OPT - izbacuje stranicu koja nece biti koriscena

FIFO

0	0	0	1	1	1	1	1	7	7	7	1
	7	7	7	7	3	3	3	3	2	2	2
		2	2	2	2	2	4	4	4	5	5
prekidi :	1	2	3	4		5		6	7	8	9	10

Imamo 10 prekida.

LRU

0	0	0	1	1	1	2	2	2	2	2	2
	7	7	7	7	7	7	4	4	4	5	5
		2	2	2	3	3	3	7	7	7	1
prekidi:	1	2	3	4		5	6	7	8		9	10

Imamo 10 prekida.
Svejedno koji cemo algoritam koristiti.

FIFO

0	0	0	0	0	3	3	3	3	3	5	5
	7	7	7	7	7	7	4	4	4	4	1
		2	2	2	2	2	2	7	7	7	7
			1	1	1	1	1	1	2	2	2
prekidi:	1	2	3	4		5		6	7	8	9	10

Imamo 10 prekida.

LRU

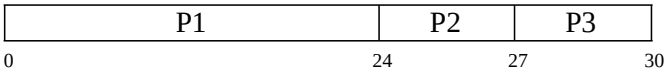
0	0	0	0	0	3	3	3	3	3	5	5
	7	7	7	7	7	7	7	7	7	7	7
		2	2	2	2	2	2	2	2	2	2
			1	1	1	1	4	4	4	4	1
prekidi:	1	2	3	4		5		6			7	8

Imamo 8 prekida.
Bolje je koristiti LRU algoritam.

1) Prikazati Gantt-ov dijagram planiranja za procese navedene u sledecoj tabeli ako se primenjuje FCFS („first-out-first-served“) algoritam.

proces	vreme izvrsavanja
P1	24
P2	3
P3	3

Pretpostavimo da procesi dolaze u sledecem redosledu: P1, P2, P3.



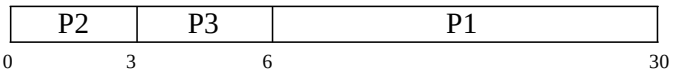
Vreme cekanja na dodelu procesora je:
P1 = 0
P2 = 24
P3 = 27

Prosecno vreme cekanja je:
(0 + 24 + 27) / 3 = 17

2) Prikazati Gantt-ov dijagram planiranja za procese navedene u sledecoj tabeli ako se primenjuje FCFS algoritam.

proces	vreme izvrsavanja
P1	3
P2	3
P3	24

Pretpostavimo da procesi dolaze u sledecem redosledu: P2, P3, P1.

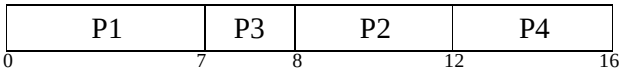


Vreme cekanja na dodelu procesora je:
P1 = 6
P2 = 0
P3 = 3

Prosecno vreme cekanja je:
(6 + 0 + 3) / 3 = 3

3) Prikazati Gantt-ov dijagram planiranja za procese navedene u sledecoj tabeli ako se primenjuje SJF algoritam bez prekidanja.

proces	vreme dolaska	vreme izvrsavanja
P1	0.0	7
P2	2.0	4
P3	4.0	1
P4	5.0	4

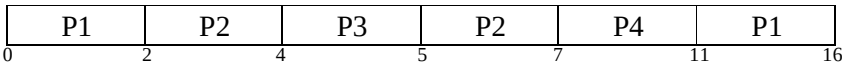


Vreme cekanja na dodelu procesora:
P1 = 0
P2 = 8 – 2 = 6
P3 = 7 – 4 = 3
P4 = 12 – 5 = 7

Prosecno vreme cekanja:
(0 + 6 + 3 + 7) / 4 = 4

4) Prikazati Gantt-ov dijagram planiranja za procese navedene u sledecoj tabeli ako se primenjuje SRTF algoritam sa prekidima.

proces	vreme dolaska	vreme izvrsavanja
P1	0.0	7
P2	2.0	4
P3	4.0	1
P4	5.0	4

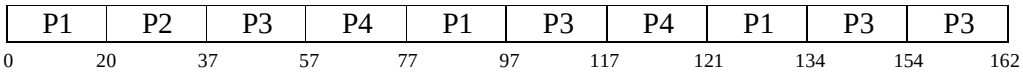


Vreme cekanja na dodelu procesora:
P1: 11 – 2 = 9
P2: 5 – 4 = 1
P3: 0
P4: 7 – 5 = 2

Prosecno vreme cekanja:
(9 + 1 + 0 + 2) / 4 = 3

5) Prikazati Gantt-ov dijagram planiranja za procese navedene u sledecoj tabeli ako se primenjuje Round-Robin algoritam sa vremenskim kvantumom 20.

proces	vreme izvrsavanja
P1	53
P2	17
P3	68
P4	24



Vreme cekanja na dodelu procesora:
P1: 20 + 20 + 20 + 20 + 20 – 3 = 100 – 3 = 97
P2: 20
P3: 20 + 17 + 40 + 4 + 13
P4: 40 + 17 + 20 + 20

1) Pet (5) procesa A,B,C,D i E dolaze u ovom redosledu istovremeno kao sto je na tablici pokazano. Manja vrednost znaci visi prioritet.

	Koriscenje Procesora	Prioritet
A	4	3
B	6	5
C	3	1
D	2	4
E	7	2

a) FCFS
b) SJF (non-preemtive)
c) Po prioritetu
d) Round Robin (time quantum = 2)

Resenje:

Scheduling policy	Waiting time					Average waiting time
	A	B	C	D	E	
FCFS	0	4	10	13	15	42/5 = 8.4
SJF (shortest job first)	5	9	2	0	15	31/5 = 5.2
Priority	10	16	0	14	3	43/5 = 8.6
Round-Robin	8	13	12	16	15	54/5 = 10.8

2) Koji su potrebni uslovi za nastanak potpunog zastoja?

- Medjusobno iskljucenje (“mutual exclusion”)
- Posedovanje i cekanje (“hold-and-wait”)
- Kruzno cekanje (“circular wait”)
- Nema prekidanja (“no preemption”)

3) Sta je inverzija prioriteta procesa? Objasniti kako nastaje.

- Ako proces viseg prioriteta zahteva pristup resursu koji je zauzet od procesa nizeg prioriteta, tada je proces viseg prioriteta blokiran procesom nizeg prioriteta, sve dok proces nizeg prioriteta ne zavrsi rad sa resursom. Na taj nacin proces nizeg prioriteta ima “visi” prioritet.

4) Kako se moze resiti problem inverzije prioriteta procesa?

- Proces nizeg prioriteta koji pristupa datom resursu nasledjuje visi prioritet procesa koji zahteva pristup istom resursu, sve do zavrsetka rada sa resursom. Na taj nacin se ubrzava izvrsavanje procesa nizeg prioriteta. Kada proces nizeg prioriteta zavrsi, njegov prioritet se vraca na originalnu vrednost. To je protokol nasledjivanja prioriteta.

5) Objasniti sta je “trashing”.

- Stanje sistema kada se mnogo vise vremena trosi na obradu stranicnih prekida nego na izvrsavanje procesa.

6) Kada sistem detektuje "trashing" kako moze da eliminise ovaj problem?

a) Povecanjem broja poslova koji intenzivno trose procesorsko vreme
b) Smanjenjem stepena multiprogramiranja - TACNO
c) Instaliranjem brzeg procesora
d) Ni jednom prethodno navedenom aktivnosti

7) Cemu sluzi stek (stack) memorija?

- Za memorisanje parametara prilikom poziva procedura i za lokalne promenljive. Drugim recima, za podatke koji se dinamicki dodeljuju po unapred definisanom redosledu (predictable order). Ovom memorijom upravlja kompajler, korisnik ne upravlja.

8) Cemu sluzi hip ("heap") memorija?

- Koristi se za proizvoljne strukture podataka, kao sto su povezane liste itd.. Drugim recima, za podatke koji se dinamicki dodeljuju na nacin koji nije unapred definisan (unpredictable manner)
- Ovom memorijom upravlja korisnik (potrebni su pozivi za alokaciju i dealokaciju memorije).

Vezbe 15.12.2007.

---

1) Pretpostavite da imate trag pristupa stranama jednog procesa kome je dozvoljeno (m) okvira strana u memoriji (koji su na pocetku izvrsavanja celi prazni). Duzina traga pristupa stranama je (p) – ukupno (p) pristupa memoriji, pri cemu broj razlicitih strana kojima se pristupa iznosi (n). Bez obzira na algoritam zamene strana, navedite:
   a) koja je donja granica broja stranicnih prekida;
   b) koja je gornja granica broja stranicnih prekida.

a) min = n - najmanji, jer se svaka razlicita strana mora „load"-ovati u glavnoj memoriji.
b) max:
   m >= n => n - jer nije potrebno da se zameni bilo koja strana;
   m < n => p - jer se za svaku stranu moze pojaviti stranicni prekid.

2) Navesti bar dva razloga zbog kojih se koristi sistem virtuelne memorije.

- pojednostavljuje povezivanje (linking) i punjenje (loading);
- svakom procesu je omoguceno da ima citav adresni prostor i da moze da koristi bilo koju adresu unutar tog adresnog prostora.
- zastita – onemogucava jednom procesu da pristupa memoriji drugog procesa.

3) Opisati kako se virtuelna adresa prevodi u fizicku adresu.

4) Koja je namena TLB bafera („Translation tookaside buffer")?

- TLB je namenski kes za podatke iz tabele stranica (u ovom baferu kesiraju se samo podaci iz tabele stranice).

5) Ukratko porediti performanse sistema virtuelne memorije sa i bez TLB bafera.

- Bez TLB, svaki pristup procesora (CPU) memoriji zahteva dva pristupa fizickoj memoriji – jedan za podatke u tabeli stranica, a drugi pristup potrebnim podacima koji se nalaze u rezultujucoj stranici.

- Sa TLB, moze se ocekivati broj pristupa fizickoj memoriji koji je vrlo blizu broja 1.

6) Koliko je vrsta potrebno u tabeli stranica (za 1 proces) u sistemu virtuelne memorije koji ima 16-bitne virtuelne adrese („byte-adresses“ – memorijske lokacije velicine 1 bajta) i svaka stranica je velicine 1024 bajta?

$1024 = 2^{10}$

$2^{16} / 2^{10} = 2^6 = 64$ vrste

7) Dati sistem ima 48-bitne virtuelne adrese, 36-bitne fizicke adrese i 128 MB glavne memorije. Ako sistem koristi stranice velicine 4096 bajtova (4 KB), koliko virtuelnih i fizickih stranica adresni prostori mogu da podrze? Koliko ima okvira strana u glavnoj memoriji?

$4096 = 2^{12}$ => 12 bita za pomeraj

a) $2^{(48-12)} = 2^{36}$ i $2^{(36-12)} = 2^{24}$

b) 128 MB / 4096 B = 2^{20} * 128 B / 2^{12} B = 128 * 2^8 = 32 768 okvira strana
(1 MB = 2^{10} KB = 2^{20} B)

8) Dati sistem ima 64-bitne virtuelne adrese, 32-bitne fizicke adrese i 2 GB glavne memorije. Ako sistem koristi stranice velicine 4096 bajtova, koliko virtuelnih i fizickih stranica adresni prostori mogu da podrze? Koliko ima okvira strana u glavnoj memoriji?

a) $2^{(32-12)} = 2^{20}$ fizickih stranica

$2^{(64-12)} = 2^{52}$ virtuelnih stranica

b) $2^{20} * 2 / 4 = 2^{19}$ okvira stranica

9) Dati sistem ima 32-bitne virtuelne adrese, 32-bitne fizicke adrese i stranice velicine 4096 bajtova. Dat je, takodje, sledeci skup preslikavanja adresa:

Broj virtuelne strane	Broj fizicke strane
0x abc89	0x 97887
0x 13385	0x 99910
0x 22433	0x 00001
0x 54483	0x 1a8c2

Koje fizicke adrese odgovaraju sledecim virtuelnim adresama:
a) 0x 22433007
b) 0x 13385abc

c) 0x abc89011

a) Virtuelne adrese prebaciti u fizicke adrese:
224 33 007
32

0 x 007 – pomeraj, 0 x 0001007
Fizicka adresa
ostaje: 22433 -> 0x0001

b) 13385abc
32

0x abc – pomeraj => 0x99910abc
13385 -> 0x 9910
nova fizicka adresa

c) abc89011
32

0x 011 – pomeraj
abc89 -> 0x 97887 => 0x97887011
nova fizicka adresa

10) Dati sistem ima 32-bitne virtuelne adrese, 24-bitne fizicke adrese i stranice velicine 2048 bajtova.
   a) Kolika je velicina jednog sloga u tabeli stranica?
   b) Koliko je slogova u tabeli stranica potrebno za ovaj sistem?
   c) Koliko je memorijskog prostora potrebno za tabelu stranica?

32 VA
24 FA
$2048 = 2^{11}$ (11 mesta za pomeraj)

a) 24 – 11 = 13 bita za fizicku stranicu
   13 + 2 = 15 ~ 16 bita = 2 B
   (napomena: da je 17 bilo bi priblizno 32!)

b) 32 – 11 = 21 => 2^{21} slogova – preko virtuelne

c) $2 * 2^{21} = 2^{22}$ memorijskog prostora
   2B

11) Dati sistem ima 32-bitne virtuelne adrese i stranice velicine 4KB. Velicina jedne vrste (page table entry) tabele stranica je 4 bajta. Kolika je velicina tabele stranica?

$2^{(32-12)} = 2^{20}$
$2^{20} * 4 = 2^{22} = 4$ MB

12) Dati sistem ima 32-bitne virtuelne adrese i stranice velicine 4KB. Velicina fizicke memorije je 512 MB. Velicina jedne vrste tabele stranica je 8 bajtova (4 bajta su predvidjena za virtuelnu adresu). Kolika je velicina invertovane tabele stranica?

32 VA
4 KB = 2^{12}
512 MB
8 B = 2^{3}

- Ide se preko fizicke adrese.
$(512 / 2^{12}) * 2^{3} = 2^{20} = 1$ MB

13) Sistem raspolaze sa 12 primeraka jednog resursa. Trenutno su aktivna tri procesa P1, P2 i P3 sa stanjem prikazanim u sledecoj tabeli:

proces	max. zahteva	trenutno dodeljeno
P1	10	5
P2	4	2
P3	9	3

Da li se sistem nalazi u bezbednom, nebezbednom ili u stanju zastoja? Obrazloziti odgovor.

U nebezbednom stanju je.
Imamo 12 resursa, a trenutno su dodeljena 5 + 2 + 3 = 10. Ostaju dva slobodna resursa i moze da se zavrsi proces P2 koji zahteva jos dva resursa. Posle njegovog izvrsenja oslobadja se 4 resursa, ali P1 zahteva jos 5, a P3 jos 6 resursa. Ukoliko nijedan od procesa ne oslobodi resurse, dolazi do zastoja.

14) Drajver diska ima zahteve za sledece cilindre 10, 22, 20, 2, 40, 6 i 38 i to navedenim redosledom. "Seek time" po cilindru je 6 msec. Koliko je "seek time"-a potrebno ako se primenjuje:
   a) FIFO algoritam?
   b) SSTF algoritam (da se glava diska najmanje zavrti)
   (u svim slucajevima glava diska je inicijalno na cilindru 20)

a) $20 \xrightarrow{10} 10 \xrightarrow{12} 22 \xrightarrow{2} 20 \xrightarrow{18} 2 \xrightarrow{38} 40 \xrightarrow{34} 6 \xrightarrow{32} 38$

(12 + 2 + 18 + 38 + 34 + 32) * 6 = 146 * 6 = 876 msec

b) $20 \xrightarrow{0} 20 \xrightarrow{2} 22 \xrightarrow{12} 10 \xrightarrow{4} 6 \xrightarrow{4} 2 \xrightarrow{36} 38 \xrightarrow{2} 40$

(0 + 2 + 12 + 4 + 4 + 36 + 2) * 6 = 60 * 6 = 360 msec

15) Dat je sistem stranicenja koji koristi 16-bitne adrese. Velicina stranica je 4 KB, broj slogova u tabeli stranica je 4 i sistem ima 8 okvira stranica u fizickoj memoriji. Na sledecoj slici su prikazane tabele stranica za dva procesa koji se izvrsavaju, P1 i P2.

Potrebno je odrediti fizicke adrese za:

a) logicku adresu 15000 procesa P1;
b) logicku adresu 12000 procesa P2.

P1

0
4
5
2

P2

3
1
7
6

a) 15000 – 3 * 4096 = 2712 <- pomeraj unutar stranice 3
fizicka adresa: 2 * 4096 + 2712 = 10904

b) 12000 – 2 * 4096 = 3808
fizicka adresa: 7 * 4096 + 3808 = 32480

Vezbe 25.12.2007.

---

1. Operativni sistemi upraljavljaju samo hardverom?

- Ne. OS upravljaju i izvrsavanjem aplikacija i drugog softvera.

2. Zasto je teze projektovati OS danas u odnosu na pre 50 godina?

- OS pre 50 god. su upravljali malim brojem programa i hardverskih uredjaja.

3. Zasto je BIOS vazan za racunarske sisteme?

- BIOS izrazava osnovnu inicijalizaciju hardvera i upravlja i puni osnovne komponenete OS u memoriju. Takodje, obezbedjuje instrukcije koje omogucavaju OS da komunicira sa hardver sistemom.

4. Zasto OS treba da spreci korisniku da pristupe “boot” sektoru?

- Ako bi korisnici mogli da pristupe “boot” sektoru tada bi oni mogli greskom ili namerno da modifikuju kod OS-a i na taj nacin ucine OS neupotrebljivim ili da omoguce napadacu da preuzme kontrolu nad datim sistemom.

5. Da li softver napisan u masinskom jeziku moze da se izvrsava na razlicitim masinama?

- Masinski jezici su zavisni od masina tako da softver moze da se izvrsava u masinskom jeziku samo na masinama istog tipa.

6. Da li se asemblerski kod izvrsava direktno?

- Netacno. Asembleri prevode asemblerski kod u kod masinskog jezika pre nego sto se kod moze izvrsiti.

7. Zasto su programi prevedeni u "BYTECODE"portabilniji od programa prevedenim u masinski kod?

- "BYTECODE"je pripremljen za izvrsavanje na virtualnoj masini koja moze biti instalirana na vise razlicitih platformi.

8. Koje komponente OS izvrsavaju sledece operacije:
a) pisanje po hard disku
b) odredjivanje koji ce se sledeci process izvrsavati
c) odredjivanje gde ce se u memoriji smestiti novi proces
d) organizovanje datoteka na disku.
e) omogucava jednom procesu da posalje podatke nekom drugom procesu.

a) Input/Output maneger
b) Processor scheduler
c) Memory maneger
d) File system menager
e) Interprocess communication(IPC maneger)

9. Koja je glavna karakteristika monolitnog OS-a?

- Kod monopolitnog OS-a svaka komponenta OS-a se nalazi unutar kernela.

10. Zasto su viseslojni(visenivosni) OS-a modularniji od monolitnih OS-a?

- Kod viseslojnih OS-a implentacija i interfejsi su razdvojeni za svaki sloj(nivo). To omogucava da se svaki sloj(nivo) testira i debagira posebno. Takodje omogucava projektantima da promene implementaciju svakog sloja bez potrebe da se modifikuju drugi slojevi.

11. Zasto su visenivoviski sistemi manje efikasni od monolitnih OS-a?

- Kod visenivovskih OS-a nekoliko poziva moze biti potrebno za komunikaciju izmedju slojeva, dok ovaj "overhead" ne postoji kod monolitnih OS-a.

12. Koja je razlika izmedju visenivoske arhitektura i microkernel arhitekture?

- Visenivoska arhitektura omogucava ekskluzivnuo komunikaciju izmedju komponenti OS-a u susednim slojevima(nivoima).
- Mikrokernel arhitektura omogucava komunikaciju izmedju komponenti OS-a preko mikrokernela.

13. Termin process i program su sinonimi?

- Netacno. Proces je program koji se izvrsava, a program je staticki element.

14. Koja je namena tabele procesa?

- Tabela procesa omogucava OS-u da za svaki process locira kontrolni blok procesa (PCB). Tabela procesa sadrzi identifikator procesa (PID) i pokazivac na kontrolni blok procesa.

15. Struktura kontrolnog bloka procesa(PCB) je zavisna od implementacije OS-a?

- Tacno. Proces koji se prvo kreira na Unix sistemima i koji se zove init nema roditeljski process. Takodje, na nekim sistemima kada se unisti process roditelj njegova deca nastavljaju sa izvrsavanjem.

16. Kako drajveri uredjaja i "plug and play" interfejsi uticu na prosirenje OS-a?

- Drajveri uredjaja omogucavaju OS-u da prepozna hardverski komponente i da upravlja njima, a "plug and play" interfejsi olaksavaju instalaciju.

Predavanja 05.01.2008.

---

1) OPT algoritam zamene strana

0 3 2 7 4 2 1 0 5 3

	0	0	0	0	0	0	0	0	5	5
		3	3	3	3	3	3	3	3	3
			2	2	2	2	1	1	1	1
				7	4	4	4	4	4	4
prekidi:	1	2	3	4	5		6		7	

Imamo 7 stranicnih prekida.

2) Kako se upisuju datoteke na CD-u?

- Datoteke se na CD-u upisuju kontinualno. Za pristup se koristi drajver CD-a. Datoteteke se skladiste sekvencijalno.

3) Navesti aktivnosti servisiranja stranicnog prekida.

-1. Utvrditi da li je adresa kojoj se pristupa u memoriji. Ako jeste nastavi sa izvrsavanjem instrukcije. U suprotnom idi na korak 2

2.Prekini izvrsavanje programa

3. Nadji slobodan okvir u memoriji. Ako takav okvir ne postoji izbaci jednu od strana,odnosno oslobodi jedan od okvira koji su dodeljeni programu

4. Pronadji na disku stranu kojoj se pristupa i upisi je u slobodan okvir

5. Azuriraj tabelu strana

6. Iniciraj izvrsavanje instrukcije koja je izazvala prekid.

POGLAVLJE 5: OPERATIVNI SISTEMI

1) Sta je operativni sistem?

- Operativni sistem je sistemski softver koji upravlja dodelom i efikasnim koriscenjem resursa datog racunarskog sistema potrebnih za resavanje datog problema.

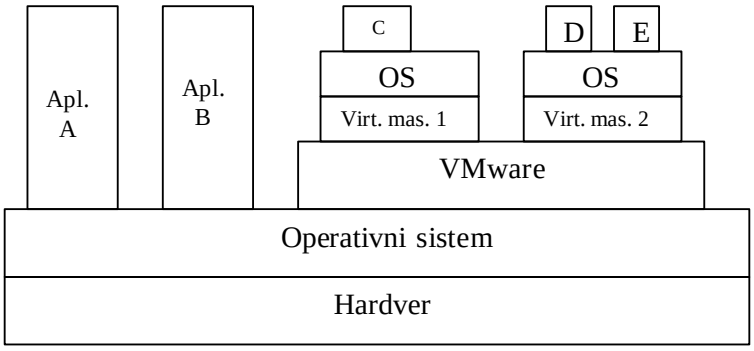
2) Navesti primere poznatih operativnih sistema.

- Primeri operativnih sistema su: UNIX, Linux, BeOS, QNX, iRMX, VRTX32, VCOS, Mac OS, DOS, Microsoft Windows 95/98/XP/NT/2000/2003/Vista, Solaris, Symbian OS, Palm OS itd.

3) Objasniti zasto je operativni sistem reaktivni program.

- Cest je slucaj da aplikacija pristupa hardveru preko operativnog sistema. U bilo kom trenutku na sistemima koji imaju samo jedan centralni procesor izvrsava se samo jedna aplikacija. Ponekad je to operativni sistem, a nekad korisnicka aplikacija. Kada se izvrsava korisnicka aplikacija OS gubi svoju kontrolu nad masinom. OS je reaktivni program i on ponovo dobija kontrolu kada korisnicka aplikacija izvrsi sistemski poziv ili ako se pojavi hardverski prekid. Drugim recima, OS je konstantno u stanju cekanja da se desi jedan od sledeca dva tipa dogadjaja: prekid ili sistemski poziv.

4) Prikazati primer kreiranja vise virtuelnih masina nad istim harverom i operativnim sistemom.



5) Koje su moguce vrste interfejsa izmedju krajnjeg korisnika i operativnog sistema?

- Interfejs moze biti GKI (graficki korisnicki interfejs), bilo koja aplikacija ili interpreter komandi.

6) Na koji nacin se operativni sistem "budi" iz stanja praznog hoda?

- Iz stanja praznog hoda operativni sistem se "budi":
  - prekidom dobijenim od nekog hardverskog uredjaja
  - pojavom izuzetka od nekog korisnickog programa
  - sistemskim pozivom iz nekog korisnickog programa

7) Koji su osnovni ciljevi savremenih operativnih sistema?

- Osnovni ciljevi savremenih operativnih sistema su:
  - da olaksa izvrsavanje aplikativnih programa i resavanje korisnickih problema
  - da ucini racunarski sistem jednostavnim i podesnim za koriscenje
  - da obezbedi deljenje racunarskih resursa
  - da omoguci koriscenje racunarskog harvera na efikasan nacin

8) Dati primer apstrakcije harvera racunarskih sistema.

Apstrakcija hardvera racunarskih sistema

HARDVER
Diskovi
Memorija
Procesori
Mreza
Modem
Monitor
Tastatura
Mis

9) Dati primer apstrakcije ostalih resursa racunarskih sistema koji nisu harver.

Apstrakcija ostalih resursa racunarskih sistema

RESURSI
Datoteke
Programi
Niti/Procesi
Komunikacija
Prozori i GKI

10) Cemu sluzi poznavanje rada savremenih operativnih sistema?

- Pravi se razlika izmedju nekog ko se profesionalno bavi racunarskim sistemima i nekog ko je samo krajnji korisnik. Razumevanje rada operativnih sistema omogucava uspesniju primenu racunarskih sistema i predstavlja vazan preduslov za poboljsanje performansi sirokog opsega aplikacija kod kojih se u toku zivotnog ciklusa moze pojaviti problem performansi.

11) Koji su kljucni aspekti svakog operativnog sistema?

- Kljucni aspekti svakog operativnog sistema su:
  a) apstrakcija resursa i
  b) deljenje resursa.

12) Koja su dva osnovna nacina deljenja resursa u racunarima? Navesti primere.

- Deljenje resursa u racunarima moze biti na osnovu prostornog multipleksiranja i vremenskog multipleksiranja. Primeri za prostorno mulitpleksiranje su: glavna memorija i disk. Primer deljenja resursa sa vremenskim mulitpleksiranjem u racunaru je centralni procesor.

13) Navesti servise koje obezbedjuju savremeni operativni sistemi.

- Savremeni operativni sistemi obezbedjuju sledece servise:
    1. izvrsavanje programa
    2. U/I operacije
    3. komunikacije
    4. upravljanje sistemom datoteka
    5. detekcija gresaka

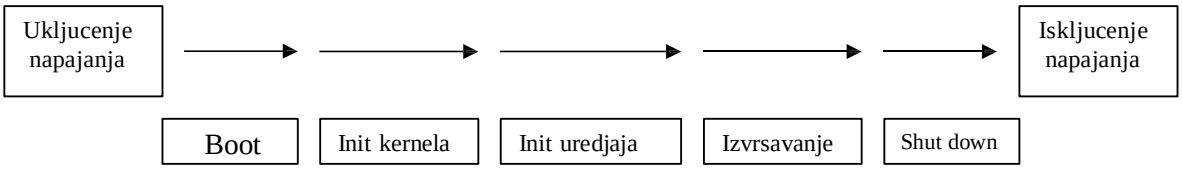
14) Koje su osnovne funkcije savremenih operativnih sistema?

- Osnovne funkcije savremenih operativnih sistema su:
    1. upravljanje procesima;
    2. upravljanje memorijom;
    3. upravljanje uredjajima;
    4. upravljanje podacima;
    5. zastita;
    6. komunikacija sa drugim racunarima u mrezi;
    7. upravljanje greskama i oporavak sistema.

15) Loader je program koji vrsi punjenje drugih programa u memoriju I prenosi kontrolu na te izvrsne programe. Ko puni loader u memoriju?

- Program koji puni "prvi program", tj. kernel u memoriju se zove bootstrap loader ili bootloader. Zahteva podrsku firmvera i obicno ima dva dela: primarni i sekunarni.

16) Graficki prikazati zivotni ciklus sistema koji obuhvata ukljucenje i iskljucenje napajanja.



17) Graficki prikazati punjenje bootloader-a i kernel-a u operativnu memoriju.

Strana 127.

18) Zasto OS treba da spreci korisniku da pristupe "boot" sektoru?

- Ako bi korisnici mogli da pristupe "boot" sektoru tada bi oni mogli greskom ili namerno da modifikuju kod OS-a i na taj nacin ucine OS neupotrebljivim ili da omoguce napadacu da preuzme kontrolu nad datim sistemom.

19) Klasifikovati operativni sisteme po broju podrzanih procesa i procesora.

Tip. OS-a	Broj procesora	Broj procesa	Deljena memorija
Monoprogramski	1	1	-
Viseprogramski	1	≥ 1	-
Viseprocesorski	≥ 1	≥ 1	Da
Distribuirani	≥ 1	≥ 1	Ne

20) Na koji nacin se kod viseprogramskih operativnih sistema povecava iskoriscenje procesora?

- Kod ovih sistema postoji preklapanje zahteva za procesorom i U/I uredjajima od strane vise programa ili korisnika.

21) Navesti nacine rada operativnih sistema na osnovu nacina interakcije korisnika sa racunarom i vremena odziva racunara na zahtev korisnika.

- Razlikujemo:
    a) paketnu obradu (batch obrada);
    b) obradu sa deljenjem procesorskog vremena (time-sharing);
    c) obradu u realnom vremenu.

22) Objasniti nacin rada operativnih sistema sa paketnom obradom.

- Kod paketne obrade poslovi se izvrsavaju onim redosledom kojim pristizu u red poslova spremih za izvrsavanje. Ovakav nacin rada bio je karakteristican za vreme kada su ulazni uredjaji bili citaci kartica i uredjaji za rad sa magnetnim trakama. Nedostatak ovakve obrade je da neki kratki poslovi mogu puno vremena da cekaju zavrsetak prethodno unetih poslova.

23) Objasniti nacin rada OS-a sa deljenjem procesorskog vremena ("time-sharing").

- Svakom procesu (korisniku) se dodeljuje procesor u unapred definisanom kvantumu vremena. Nakon isteka kvantuma operativni sistem oduzima procesor od datog procesa i dodeljuje ga sledecem procesu (korisniku).

24) Objasniti nacin rada OS-a za rad u realnom vremenu.

- Obrada u realnom vremenu je potrebna kod svih sistema kod kojih se vreme odgovora zahteva u strogo definisanim granicama. Kod sistema za rad u realnom vremenu nije dozvoljeno prekoracenje maksimalno dozvoljenog vremena. Maksimalno vreme odgovora sistema mora biti u fiksnom vremenu. Kljucni parametar koji karakterise ove sisteme je vreme.

25) Koje dve grupe operativnih sistema za rad u realnom vremenu postoje?

- Postoje:
    a) hard operativni sistem za rad u realnom vremenu (ako se data akcija mora obaviti u datom trenutku, sa zadatim vremenskim okvirom);
    b) soft operativni sistem (prihvatljiv je izostanak aktivnosti koja se mora uraditi u datom trenutku).

26) Koji tipovi ili vrste operativnih sistema postoje danas?

- Razlikujemo sledece tipove OS-a:
    1. “batch” sisteme – paketna obrada;
    2. interaktivne sisteme (jednokorisnicki i visekorisnicki);
    3. sisteme opste namene;
    4. mrezne operativne sisteme;
    5. distribuirane sisteme;
    6. specijalno projektovane operativne sisteme – sistemi za rad u realnom vremenu.

27) Kako su napisani monolitni operativni sistemi?

- Napisan je kao skup procedura pri cemu svaka od njih moze pozvati bilo koju drugu proceduru iz tog skupa kad god je to potrebno. Svaka procedura u sistemu ima dobro definisan interfejs u obliku parametra i rezultata i svaka procedura je vidljiva svakoj drugoj proceduri.

28) Navesti prednosti operativnih sistema koji imaju mikro jezgra.

- Osnovne prednosti OS-a sa mikro jezgrom su:
    1. dodavanje novog servisa ne zahteva modifikovanje jezgra operativnog sistema;
    2. sistem je bezbedniji, tj. visi je nivo zastite, jer vise operacija se izvrsava u korisnickom nacinu rada;
    3. omogucava uniformni interfejs, prosirivost, fleksibilnost i portabilnost;
    4. predstavlja podrsku distribuiranim sistemima;
    5. predstavlja podrsku objektno orjentisanim sistemima;
    6. omogucava jednostavnije projektovanje jezgra i pouzdaniji OS.

29) Na koje module se mogu dekomponovati savremeni operativni sistemi?

- Savremeni operativni sistem se moze dekomponovati u module koji obezbedjuju sl. funkcije:
    1. upravljanje procesima;
    2. upravljanje memorijom;
    3. upravljanje ulazno/izlaznim uredjajima;
    4. upravljanje datotekama;
    5. upravljanje mrezom.

30) Prikazati strukturu modularnog operativnog sistema.

Strana 134.

31) Koje su prednosti modularnih operativnih sistema?

- Postoji nekoliko prednosti modularnih sistema:
    a) lakse je modifikovati sistem i ispravljati greske, jer promene uticu samo na neke delove sistema, a ne i na ceo operativni sistem;
    b) informacije se cuvaju samo tamo gde je to potrebno;
    c) informacijama se pristupa samo unutar definisane i ogranicene oblasti, tako da bilo koji "bug" u tom delu je limitiran specificnim modulom.

32) Objasniti razliku izmedju mreznih operativnih sistema i konvencionalnih operativnih sistema koji se izvrsavaju nad jednim procesorom.

- Razlika je u dodatku kontrolera za mrezni interfejs, kao i programa za daljinsko prijavljivanje i daljinski pristup datotekama kod mreznih operativnih sistema.

33) Objasniti razliku izmedju distribuiranih operativnih sistema i konvencionalnih operativnih sistema koji se izvrsavaju nad jednim procesom.

- Bitna razlika je u mogucnosti paralelizacije izvrsavanja aplikacije u korist distribuiranih sistema. Jedan od pravaca daljeg razvoja distribuiranih operativnih sistema je implementacija poboljsanih, kao i novih tehnika i mehanizama zastite.

POGLAVLJE 6 - UPRAVLJANJE PROCESIMA

1) Sta je proces i koje su komponente procesa?

   - Proces je osnovna jedinica izvrsavanja u operativnom sistemu. Proces je dinamicki entitet koji izvrsava program nad datim skupom podataka koristeci resurse sistema. Komponente procesa su:
     a) kod programa koji se izvrsava;
     b) podaci koje program koristi;
     c) resursi potrebni programu;
     d) status procesa.

2) Ko moze incirati kreiranje procesa?

   - Proces nastaje na osnovu zahteva za izvrsavanje datog programa. Zahtev moze biti od: korisnika, operativnog sistema ili nekog drugog vec aktivnog procesa.

3) Objasniti razliku izmedju programa i procesa.

   - Izmedju programa i procesa postoji bitna razlika. Program je staticki entitet koji se sastoji od programskih instrukcija koje kada se izvrsavaju nad nekim skupom podataka definisu proces. Proces je dinamicki entitet koji izvrsava program nad datim skupom podataka koristeci resurse sistema.

4) Gde operativni sistem cuva podatke o svim procesima?

   - Operativni sistem kreira i azurira internu strukturu podataka za svaki proces. Tabela u kojoj se nalaze podaci o svim procesima zove se tabela procesa.

5) Sta je deskriptor procesa (ili kontrolni blok procesa – KBP) i koje podatke sadrzi?

   - Deskriptor procesa je skup podataka o aktivnom procesu o kojima OS vodi racuna (ime procesa, identitet vlasnika, prioritet, PSW, oblast programa, oblast podataka, vrednost registara, logicko stanje itd.).

6) Kada se kreira i kada se unistava kontrolni blok procesa?

   - Formira se u toku kreiranja procesa, a unistava po zavrsetku procesa.

7) Objasniti sta je nit (“thread”).

   - “Thread” ili nit je entitet koji se izvrsava koristeci program i druge resurse od pridruzenog procesa.

8) Koliko procesa moze biti aktivno u jednom trenutku na datom racunarskom sistemu? Obrazloziti odgovor.

   - U datom trenutku samo jedan proces moze da se izvrsava iako vise procesa moze biti spremno za izvrsavanje. Dispecar je vazan deo OS-a koji vodi racuna o rasporedjivanju procesa. Postoji vise algoritama na osnovu kojih se vrsi rasporedjivanje procesa.

9) Koja su moguca stanja procesa?

- Moguca stanja procesa su sledeca:
  1) NOV – proces je kreiran;
  2) IZVRSAVA SE – instrukcije datog programa se izvrsavaju;
  3) CEKA – proces ceka da se neki dogadjaj dogodi;
  4) SPREMAN – proces ceka da bude dodeljen procesoru
  5) ZAVRSEN – proces je zavrsio izvrsavanje.

10) Objasniti prelaske izmedju mogucih stanja procesa.

- Nakon kreiranja procesa, proces se nalazi u stanju NOV. Kada se procesu dodele potrebni resursi, osim procesora, tada on prelazi u stanje SPREMAN. Nakon dodele procesora prelazi u stanje IZVRSAVA SE. Po isteku dodeljenog kvantuma vremena proces prelazi iz stanja IZVRSAVA SE u stanje SPREMAN. Ako u stanju IZVRSAVA SE mora da saceka neki dogadjaj (npr. zavrsetak neke U/I aktivnosti), tada proces prelazi u stanje CEKA i o tome obavestava operativni sistem. Po zavrsetku U/I aktivnosti proces iz stanja CEKA prelazi u stanje SPREMAN. Proces prelazi iz stanja SPREMAN u stanje IZVRSAVA SE kada procesor postane raspoloziv i ako u toku izvrsavanja procesa se zavrsi izvrsavanje svih programski instrukcija tada proces prelazi u stanje ZAVRSEN.

11) Navesti moguce razloge blokiranja procesa.

- Moguci razlozi blokiranja procesa su: cekanje na dobijanje nekog uredjaja, upravljanje stranicnim prekidom, cekanje ulaznih podataka koje unosi korisnik preko terminala, cekanje na dozvolu ulaska u kriticnu sekciju, cekanje na unapred definisano vreme izvrsavanja dela koda, prenos podataka sa hard diska, memorije ili nekog drugog uredjaja.

12) Prikazati dijagram stanja procesa.

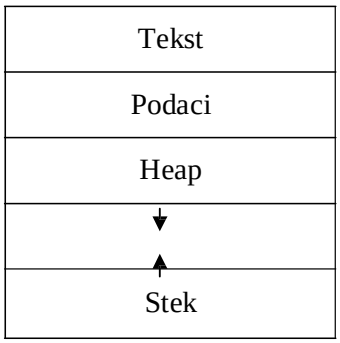
Strana 143.

13) Na koje nacine proces moze biti suspendovan?

- Proces moze biti suspendovan na zahtev korisnika, na zahtev operativnog sistema radi izbegavanja potpunog zastoja (“deadlock”) ili ako se prekoraci maksimalni dozvoljeni broj procesa u stanju SPREMAN.

14) Graficki prikazati i objasniti adresni prostor procesa.

- Svaki proces u toku izvrsavanja zahteva memoriju i zato se svakom procesu pridruzuje memorijski adresni prostor. U delu memorije koji je oznacen sa Tekst se nalaze instrukcije programa. Taj deo je fiksne velicine i instrukcije su u izvrsnom obliku, tj. masinske instrukcije programa koji se izvrsava. U delu memorije koji je oznacen sa Podaci se nalaze predifinisane strukture podataka koje su inicijalizovane pre pocetka izvrsavanja programa. Treci deo oznacen sa Heap se koristi za dinamicku dodelu memorije. Ovaj deo je promenljive velicine. U dela Stek se nalaze izvrsni okviri funkcija koje su pozvane od strane datog programa. Povecanje i smanjenje velicine dela Stek je automatski.



15) Koja su moguca stanja niti?

- Moguca stanja niti su: STANJE IZVRSAVANJA, STANJE SPREMNOSTI, STANJE CEKANJA.

16) Prikazati dijagram stanja niti.

Strana 148.

17) Koji se resursi koriste kada se kreira nit? U cemu je razlika u odnosu na resurse koji se koriste kada se kreira proces?

- Za kreiranje niti potrebno je manje resursa nego za kreiranje procesa Kreiranje procesa zahteva dodelu kontrolnog bloka procesa (KBP), tj. prilicno velike strukture podataka. Kreiranje bilo sistemske bilo korisnicke niti podrazumeva dodelu male strukture podataka u kojoj se cuvaju sadrzaji skupa registara, sadrzaj steka i prioritet.

18) Navesti primere aplikacija koje koriste vise niti i na taj nacin ostvaruju poboljsanje performansi u odnosu na resenje sa jednom niti.

- Primer aplikacije sa vise niti je Web server koji svaki zahtev servisira u posebnoj niti. Drugi primer je interaktivni program sa grafickim korisnickim interfejsom (npr. debugger), gde se jedna nit koristi za nadgledanje korisnickih ulaznih podataka, druga nit predstavlja aplikaciju koja se izvrsava i treca nit moze da nadgleda performanse.

19) Graficki prikazati blok semu rada operativnog sistema sa redovima cekanja.

Strana 149.

20) Objasniti prelaske procesa izmedju razlicitih redova cekanja u toku svog zivotnog ciklusa.

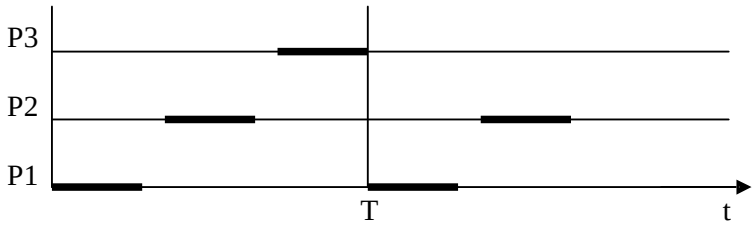
- Najpre se svaki proces nalazi u redu cekanja poslova. Nakon prebacivanja u glavnu memoriju i dodele potrebnih resursa proces prelazi u red cekanja spremnih poslova. Kada proces dobije procesor moguci su sledeci slucajevi:
a) proces se izvrsava sve dok ne istekne dodeljeno vreme;
b) postoji zahtev za izvrsavanje U/I operacije i proces se prebacuje u U/I red cekanja i
c) nastaje prekid i proces prelazi u stanje spreman.

21) Objasniti strukturu podataka koja predstavlja posao, tj. proces.

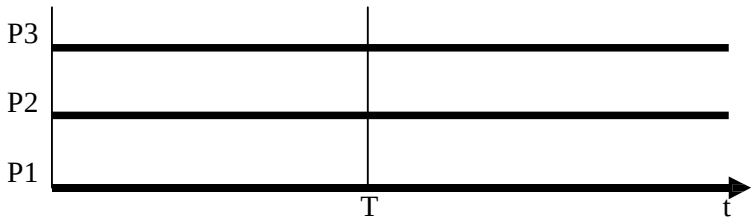
Vlasnik
Jedinstven identifikator procesa
Opis adresnog prostora
Pocetna adresa programa
Pocetna adresa podataka
Pocetna adresa “stack”-a
Lista niti
Broj niti
Informacije o otvorenim datoteka.
“Default” direktorijum

Struktura podataka koja predstavlja posao, tj. posao

22) Prikazati kvaziparalelno izvrsavanje programa na primeru programa P1, P2 i P3 pod uslovom da programi imaju isti prioritet izvrsavanja i da operativni sistem radi u “time-sharing”-u.



23) Prikazati paralelno izvrsavanje programa na primeru programa P1, P2 i P3.



24) Objasniti sta je kriticna sekcija za vise niti ili procesa.

- Kriticna sekcija je segment koda cije instrukcije mogu da uticu na druge niti (npr. azuriranje vrednosti zajednicke promenljive). Kada jedna nit izvrsava kriticnu sekciju ni jedna druga nit ne sme da izvrsava tu istu kriticnu sekciju.

25) Sistem se sastoji od dva procesa S i R koji dele zajednicki bafer kapaciteta C poruka. Proces S upisuje poruke u bafer, a proces R ih cita iz bafera. Koji uslovi moraju biti ispunjeni da bi sistem ispravno funkcionisao u situaciji kada relativne brzine pristupa baferu procesa S i R nisu unapred poznate?

- Da bi procesi S i R ispravno funkcionisali, u situaciji kada njihove relativne brzine nisu unapred poznate, neophodno je da budu ispunjeni sledeci uslovi:
1) posaljilac S ne sme da salje poruke kada je bafer pun, jer bi se ona izgubila;
2) primalac R ne sme da uzima poruku kada je bafer prazan, jer bi to mogao protumaciti kao signal da treba da prestane sa radom;
3) procesi ne smeju istovremeno pristupiti baferu;

4) vrednost promenljive N mora, u trenutku pristupa baferu bilo kog procesa, biti jednaka stvarnom broju poruka u baferu.

29) Navesti osnovne zahteve koje OS mora da obezbedi u cilju uspesnog upravljanja konkurentnim procesima.

- Zahtev 1. Uporedni procesi se kriticnim sekcijama, jedan u odnosu na drugog, moraju se medjusobno iskljucivati tako da se istovremeno ne nalaze u svojim kriticnim sekcijama.
- Zahtev 2. Proces koi se zaustavio van svoje kriticne sekcije ne sme onemoguciti dalje odvijanje drugih nezavisnih procesa.
- Zahtev 3. Procesi ne smeju beskonacno dugo cekati na resurse ili signale.

30) Kako je E. Dijkstra resio problem medjusobnog iskljucenja i sinhronizacije procesa?

- E Dijkstra je dao resenje problema medjusobnog iskljucenja i sinhronizacije procesa tako sto je uveo novi tip promenljive koja se naziva semfor i ciji je skup dozvoljenih vrednosti skup nenegativnih celih brojeva. Pritom je uve dve dozvoljene operacije nad promenljivim (P i V).

31) Definisati P i V operacije nad promenljivom s tipa semafor.

Strana 161.

32) Objasniti sta je zaposleno cekanje ("busy wait").

- Zaposleno cekanje je situacija kada proces koji pokusava da izvrsi P operaciju u situaciji kada je vrednost promenljive s jednaka nuli, trosi vreme centralnog procesora, pri cemu ne moze da napreduje dalje, a istovremeno onemogucava ostale procese da se izvrsavaju.

34) U okviru kog dela softvera se implementiraju P i V operacije nad promenljivim tipa semafor? Obrazloziti odgovor.

- P i V operacije menjaju stanja aktivnih procesa (iz IZVRSAVA SE u CEKA, odnosno iz CEKA u SPREMAN), sto znaci da se obe operacije moraju implementirati kao sastavni deo operativnog sistema. Drugim recima, procesi korisnika ne mogu direktno izvrsavati ove operacije, vec, pozivanjem ovih procedura predaju upravljanje operativnom sistemu koji ih zatim izvrsava.

36) Prikazati kako se problem medjusobnog iskljucenja i sinhronizacije posaljioca poruke S i procesa primaoca poruke R moze uspesno resiti koriscenjem semafora ako je dat bafer kapaciteta C poruka.

- Uvode se tri semafora. Prvi semafor im uzima (ima mesta) uzima vrednosti iz skupa {0, C} gde je C kapacitet bafera. Drugi semafor ip (ima poruka) uzima vrednosti iz istog skupa ali tako da je im + ip = C. Ova dva semafora omogucavaju sinhronizaciju procesa S i R, tako da proces S vrsi P operaciju nad semaforom im. Ukoliko je im>0 proces smanjuje broj mesta u baferu, upisuje poruku i vrsi V operaciju nad semaforom ip, tj. povecava broj poruka za jedan. Ukoliko je im = 0, proces S ceka da se oslobodi mesto u baferu. Slicno tome proces R izvrsava P

operaciju nad semaforom ip, cime smanjuje broj poruka u baferu, a zatim izvrsava operaciju V nad semaforom im cime povecava broj slobodnih mesta za jedan. Ukoliko je ip = 0, proces R ceka da se unese poruka u bafer. Konacno, semafor medisk sluzi za medjusobno iskljucenje procesa S i R u odnosu na pristup baferu koji je zajednicki resurs za oba procesa.

37) Na koji nacn kratkorocni planer ucestvuje u radu sa semaforima?

- U okviru operativnih sistema implementacija semafora ukljucuje interakciju sa planerom:

a) za blokiranje niti;
b) za deblokiranje niti;
c) za upravljanje redovima cekanja niti u stanju cekanja.

POGLAVLJE 7 – UPRAVLJANJE MEMORIJOM

1) Koje su najvaznije aktivnosti operativnog sistema u delu za upravljanje memorijom?

- Najvaznije aktivnosti u delu za upravljanje memorijom su:
  1) vodjenje evidencije o tome koji se delovi memorije trenutno koriste i ko ih koristi;
  2) donosenje odluke o ucitavanju procesa u memoriju, tj. koje procese prebaciti u memoriju kada memorijski prostor postane raspoloziv i
  3) dodela i oslobadjanje memorijskog prostora po potrebi.

2) Navesti komponente upravljanja memorijom pomocu kojih operativni sistem donosi bitne odluke
neophodne za efikasno upravljanje memorijom.

- Upravljanje memorijom se sastoji od sledece tri komponente:
  1) upravljanje unosenjem ("fetch policy") – u smislu donosenja odluke o tome kada ce se program ili njegovi delovi uneti u memoriju;
  2) upravljanje smestanjem ("placement policy") – u smislu donosenja odluke o tome gde ce se program ili njegovi delovi smestiti u memoriju;
  3) upravljanje zamenom ("replacement policy") – u smislu donosenja odluke o tome koji ce se program ili delovi programa izbaciti iz memorije da bi se oslobodio prostor za unosenje drugog programa ili delova drugog ili istog programa.

3) Navesti moguce nacine za upravljanje memorijom.

- Pomocu statickih particija, pomocu dinamickih particija, pomocu statickih stranica, pomocu dinamickih stranica, pomocu statickih segmenata, pomocu dinamickih segmenata i pomocu dinamickih stranica i segmenata.

4) Prikazati memorijski sistem kod savremenih racunarski sistema.

Strana 175.

5) Objasniti sta znaci vezivanje adresa ("address binding").

- Vezivanje adresa je proces pridruzivanja fizickih memorijskih adresa adresama podataka i adresama programskih instrukcija. Moze biti dinamicki i staticko.

6) Kako se vrsi staticko vezivanje adresa?

- Kod statickog vezivanja adresa nove lokacije se odredjuju pre izvrsavanja programa. Moze biti u vreme prevodjenja koda apsolutne adrese generise programski prevodilac ("compailer") ili u vreme punjenja kada apsolutne adrese generise program za punjenje ("loader").

7) Kako se vrsi dinamicko vezivanje adresa?

- Nove lokacije se odredjuju za vreme izvrsavanja programa. Kod dinamickog vezivanja adresa apsolutne adrese generise harver.

8) Na koji nacin se generisu apsolutne memorijske adrese kod dinamickog vezivanja adresa?

- Kod dinamickog vezivanja adresa apsolutne adrese generise harver.

9) Koje funkcije preslikavanja obuhvata upravljanje memorijom?

- Obuhvata sledece funkcije preslikavanja:
1) Preslikavanje imena se odnosi na preslikavanje simbolickih adresa (imena) koje programer dodeljuje promenljivama i pojednim naredbama u programu u binarne adrese koje se cesto nazivaju jedinstvenim identifikatorima ili programskim adresama.
2) Preslikavanje adresa se odnosi na preslikavanje programskih adresa u stvarne, fizicke memorijske adrese.
3) Preslikavanje sadrzaja se odnosi na preslikavanje memorijskih adresa u vrednosi (podatke) koje one sadrze.

10) Objasniti sta je preslikavanje adresa.

- Preslikavanje adresa se odnosi na preslikavanje programskih adresa u stvarne, fizicke memorijske adrese. U slucaju statickog povezivanja adresni prostor je linearan (sastoji se od niza kontinualnih adesa). U zavisnosti od nacina upravljanja memorijom takav adresni prostor moze se preslikati u odgovarajuci linearan memorijski prostor pa se preslikavanje adrese moze opisati funkcijom:

$$a' = p + a,$$

gde je a' – memorijska adresa,
p – pocetna adresa programa u memoriji,
a – programska adresa.

11) Sta je virtuelna adresa.

- Virtuelna adresa je adresa u programu i nju generise procesor.

12) Sta je fizicka adresa.

- Fizicka adresa je adresa na racunarskom harveru.

13) Dati sistem ima 32-bitne virtuelne adrese, 32-bitne fizicke adrese i stranice velicine 4096 bajtova. Dat je, takodje, sledeci skup preslikavanja adresa:

Broj virtuelne strane	Broj fizicke strane
0x abc89	0x 97887
0x 13385	0x 99910
0x 22433	0x 00001
0x 54483	0x 1a8c2

Koje fizicke adrese odgovaraju sledecim virtuelnim adresama:
d) 0x 22433007
e) 0x 13385abc
f) 0x abc89011

b) Virtuelne adrese prebaciti u fizicke adrese:
224 33 007
32

0 x 007 – pomeraj, 0 x 0001007

Fizicka adresa

ostaje: 22433 -> 0x0001

b) 13385abc
32

0x abc – pomeraj => 0x99910abc
13385 -> 0x 9910

nova fizicka adresa

c) abc89011
32

0x 011 – pomeraj
abc89 -> 0x 97887 => 0x97887011

nova fizicka adresa

14) Dati sistem ima 64-bitne virtuelne adrese, 32-bitne fizicke adrese i 1 GB glavne memorije. Ako sistem koristi stranice velicine 2048 bajtova, koliko virtuelnih i fizickih stranica adresni prostori mogu da podrze? Koliko ima okvira strana u glavnoj memoriji?

$2048 = 2^{11}$ => 11 bita za pomeraj

a) $2^{(32-11)} = 2^{21}$ fizickih stranica
$2^{(64-11)} = 2^{43}$ virtuelnih stranica

b) $2^{20} / 2 = 2^{19}$ okvira strana

16) Opisati internu fragmentaciju memorije.

- Interna fragmentacija je deo memorije unutar regiona ili stranice koja je dodeljena datom procesu i ne koristi se od strane tog procesa. Prouzrokovana je razlicitom velicinom dodeljene memorije i programa koji je ucitan u taj deo memorije. Taj deo memorije nije raspoloziv za koriscenje drugim procesima sistema sve dok dati proces ne zavrsi radom ili ne oslobodi dodeljenu memoriju.

17) Opisati eksternu fragmentaciju memorije.

- Eksterna fragmentacija je neiskoriscena memorija izmedju particija i segmenata. Ova memorija nije kontinualna, vec se sastoji iz vise manjih delova. Ne postoji kod upravljanja memorijom pomocu statickih i pomocu dinamickih stranica.

18) Opisati "First Fit" algoritam (algoritam prvog uklapanja) za dodelu memorije.

- Kod "First Fit" algoritma operativni sistem vodi tabelu slobodnog prostora uredjenu po rastucim adresama slobodnih delova memorije. Svodi se na pronalazenje prvog slobodnog prostora, racunajuci od pocetka memorije, koji je dovoljno veliki da prihvati dati program.

19) Opisati "Best Fit" algoritam (algoritam najboljeg uklapanja) za dodelu memorije.

- Kod "Best Fit" algoritma tabela slobodnog prostora uredjuje se po rastucoj velicini slobodnog prostora. Ovaj algoritam se svodi na nalazenje najmanjeg dovoljno velikog slobodnog memorijskog prostora u koji moze da se smesti novi program.

20) Opisati "Worst Fit" algoritam (algoritam najgoreg uklapanja) za dodelu memorije.

-

4) 21) Date su dinamicke memorije sledecih velicina:
100 KB
500 KB
200 KB
300 KB
600 KB
sortirane po pocetnim memorijskim adresama u rastucem redosledu. Kako ce operativni sistem da podeli ove particije sledecim procesima, cije su zahtevane velicine memorije
212 KB
417 KB
112 KB
426 KB
ako su procesi navedeni po redosledu dolaska i ako se koristi algoritam najboljeg uklapanja, a potom i algoritam prvog uklapanja.

c) best fit
212 KB <- P1 212 -> 300
417 KB <- P2 417 -> 500
112 KB <- P3 112 -> 200
426 KB <- P4 426 -> 600

d) first fit (prvo uklapanje)
212 -> 500 KB
417 -> 600 KB
112 -> 200 KB
426 -> /

22) Koje dodatne podatke ima tabela stranica osim adrese okvira?

- Tipicna tabela stranica osim adrese okvira ima sledece dodatne podatke:
1) bit koji pokazuje da li se stranica nalazi u operativnoj memoriji, tj. da li je stranici dodeljen okvir ili ne ("valid" ili "present" bit);

2) bit koji pokazuje da li je stranica u operativnoj memoriji modifikovana ili ne ("dirty: ili "modified" bit). Ako je stranica modifikovana tada mora biti upisana na disk;

3) bit koji pokazuje da li je stranica bila koriscena skoro ("referenced" ili "used" bit) ili ne;

4) dozvola pristupa koja oznacava da je stranica "read-only" ili je "read-write".

5) nekoliko bita namenjenih za stvarno adresiranje stranice u operativnu memoriju.

23) Koja je namena TLB bafera („Translation tookaside buffer")?

- TLB je namenski kes za podatke iz tabele stranica (u ovom baferu kesiraju se samo podaci iz tabele stranice).

24) Objasniti sta je stranicni prekid.

- Kada se u toku izvrsavanja programa trazi pristup adresi koja pripada strani koja nije u memoriji dolazi do prekida programa. Ova vrsta prekida naziva se stranicni prekid.

25) Objasniti kako se servisira stranicni prekid.

-1. Utvrditi da li je adresa kojoj se pristupa u memoriji. Ako jeste nastavi sa izvrsavanjem instrukcije. U suprotnom idi na korak 2

2.Prekini izvrsavanje programa

3. Nadji slobodan okvir u memoriji. Ako takav okvir ne postoji izbaci jednu od strana,odnosno oslobodi jedan od okvira koji su dodeljeni programu

4. Pronadji na disku stranu kojoj se pristupa i upisi je u slobodan okvir

5. Azuriraj tabelu strana

6. Iniciraj izvrsavanje instrukcije koja je izazvala prekid.

POGLAVLJE 8 – DODELA PROCESORA

1.Koje vrste planera postoje kod operativnih sistema?
Kratkorocni i dugorocni.

2.Koja je uloga dugorocnog planera?
Selektuje procese koji su poslati na izvrsavanje i prebacuje ih u red procesa spremnih za izvrsavanje

3. Koja je uloga kratorocnog planera?
Selektuje proces iz reda procesa spremnih za izvrsavanje i dodeljuje mu centralni procesor.

4.Objasnite razliku u ucestalosti povezivanja dugorocnog I kratorocnog planera?
Kratorocni planer se veoma cesto poziva sto nije slucaj sa dugorocnim planerom

5.Sta je okruzenje ili kontekst procesa?
Svi sistemski resursi (registri,memorija,razne tabele) cine okruzenje

6.Sta je dispecer?
Deo OS koji dodeljuje procesor procesu koji je izabran od strane kratkorocnog planera

7.Funkcije dispecera?
Promena konteksta, prelazak u korisnicki nacin rada,skok na odgovarajucu lokaciju unatar korisnickog programa radi ponovnog startovanja tog programa

8.Sta je dispecersko kasnjenje?
Vreme koje je potrebno da dispecer zaustavi jedan proces i da pokrene drugi

9.Kriterijumi koji se koriste kod algoritama planiranja dodele?
Iskoriscavanje centralnog procesora,propusnost sistema,vreme cekanja,vreme odziva,vreme procesa provedeno u sistemu

10.Algoritam dodele centralnog procesora oredjuje redosled izvrsavanja procesa u datom sistemu .Pod predpostavkom da je dato n proces cije izvsavanje treba rasporediti na jednom procesoru koliki je broj razlicitih mogucih dodela centralnog procesora?Prikazati zavisnost kao funkciju od n.
n! = n x(n-1)x (n-2)x...x2x1

11.Objasniti razliku izmedju algoritama dodele procesora sa prekidanjem(„preemptive”) i bez prekidanja(non- preemptive)
Algoritam dodele procesora sa prekidanjem omogucava da proces moze biti prekinut u toku izvrsavanja i da se procesor dodeli nekom drugom procesu.

12.Tri karakteristicna okruzenja koja se razlikuju po nacinu optimizacije dodele procesora?
Paketna obrada,interktivni rad,rad u realnom vremenu

13.Objasniti FCFS algoritam?
Svodi se na FIFO,ne dozvoljava da se prekine proces koji se trenutno izvrsava

14. Objasniti SJF algoritam?

Planer iz reda cekanja poslova spremnih za izvrsavanje bira posao koji najmanje procesorskog vremena do zavrsetka rada. Minimizira prosecno vreme provedeno u sistemu

15.Pri kojim uslovima SJF algoritam je optimalan?

Kada su poslovi raspolozivi istovremeno

16.Koji su nedostaci SJF algoritama?

Moguce gladovanje,ne moze biti implementiran u opstem slucaju ,koristi se samo u kombinaciji sa drugim aloritmima

17.Objasniti kako moze doci do gladovanja kod primene SJF algoritma?

Neki proces ne moze da dobije procesor,jer se stalno pojavljuju procesi kojima je za zavrsetak rada potrebno manje procesorskog vremena

18. Objasniti SRTF algoritam?

Moguce je prekidanje procesa koji se trenutno izvrsava, vreme izvrsavanja mora biti unapred poznato

19.Objasniti algoritam planiranja po prioritetu procesa

Svaki proces ima pridruzen prioritet ,i naredni proces za izvrsavanje je onaj proces koji imaa najvisi prioritet

20.Koji opseg vrednosti prioriteta procesa se koristi kod Linux OS?

Kod Linux OS opseg je od 0 do 99

21.Kako se može menjati prioritet kod UNIX operativnih sistema?

Sistemskim pozivom nice() se može podešavati prioritet u opsegu od -20 do +20.

22. Ko može promeniti priritet procesa kod savremenih sistema?

Prioritet procesa može biti promenjen od strane korisnika, operativnog sistema ili kombinacijom korisnika i operativnog sistema.

23.Šta je inverzija sistema? Kako može da nastane? Kako se može rešiti?

Kada je procesoru visokog prioriteta potreban resurs koji trenutno poseduje proces niskog prioriteta, on je blokira proces visokog prioriteta i on u stvari ima „viši" prioritet, jer forsira proces visokog prioriteta da čeka. Da bi se to prevazišlo, procesi niskog prioriteta nasleđuju visok prioritet sve dok rade sa tim resursima. I bivaju ubrzani.

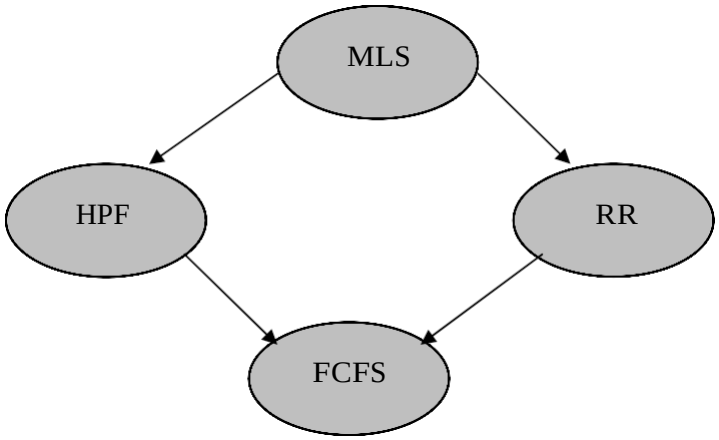
24.Objasni RR(„Round Robin") algoritam.

To je parametrizovan i kao parametar koristi interval zauzeća procesora, tj. vremenski interval ili vrem. kvantum. Za njega su svi procesi jednake važnosti.Jedan procesor dobija procesorsko vreme u trajanju jednog vremenskog intervala.

25.Objasni algoritam planiranja sa redovima čekanja u više nivoa.

U sistemu postoji više grupa procesa, za svaku grupu se formira poseban red čekanja. Planiranj se vrši po prioritetu redova ili po procentualnoj raspodeli procesorskog vremena između redova. Vremenski kvantumi su različitih vrednosti.

26. Grafički prikazati hijerarhijsko uređenje algoritma dodele procesora FCFS, RR, MLS i HPF ako je kriterijum uređenja "je specijalan slučaj".



27.Kako se određuje prosečno vreme čekanja procesa kod dodele procesora?
Kao aritmetička sredina svih vremena čekanja.

28. Prikazati Gantt-ov dijagram planiranja za procese navedene u sl.tabeli ako se privanjuju SJF algoritam bez prekida.

PROCES	VREME DOLASKA	VREME IZVREČENJA
P1	0.0	7
P2	2.0	4
P3	4.0	1
P4	5.0	4

Prosečno vreme?

P1	P3	P2	P4

0 7 8 12 16

P1=0 P3=3
P2=6 P4=7

pv= (0+6+3+7)/4 = 4

29. Prikazati Gantt-ov dijagram planiranja za procese navedene u sl.tabeli ako se privanjuju SRTF algoritam.

PROCES	VREME DOLASKA	VREME IZVREČENJA
P1	0.0	7
P2	2.0	4
P3	4.0	1
P4	5.0	4

Prosečno vreme?

P1	P2	P3	P2	P4	P1

0 2 4 5 7 11 16

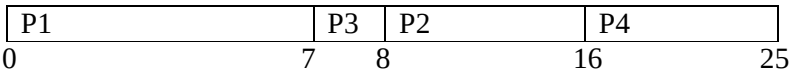
P1= 11- 2 = 9 P2 = 1 P3 = 0 P4 = 2

pv = (9+1+0+2)/4 = 3

30. Prikazati Gantt-ov dijagram planiranja za procese navedene u sl.tabeli ako se privanjuju SJF algoritam bez prekida.

PROCES	VREME DOLASKA	VREME IZVREČENJA
P1	0.0	7
P2	2.0	8
P3	4.0	1
P4	5.0	9

Prosečno vreme?



P1=0 P3=3
P2= 6 P4=11

pv= (0+6+3+11)/4 = 5

31……..

32.Objasni kako nastaje efekat konvoja kod algoritama planiranja dodele procesora.

POGLAVLJE 9 – POTPUNI ZASTOJ

1. Objasniti sta znaci potpuni zastoj(“deadlock”) ?

Odgovor: To je situacija kada proces trajno ostaje u stanju cekanja, jer u isto vreme postoje drugi procesi koji su takodje u stanju cekanja, a zauzeli su resurse koje zahteva dati process i te resurse drze zauzetim. U takvoj situaciji dati proces nikada ne dobija zahtevane resurse.

2. U uslovima viseprogramskog rada moze doci do situacije koja se naziva potpuni zastoj ili blokiranje (“deadlock”). Navesti potrebne uslove za nastanak potpunog zastoja?

Odgovor: To su: 1. Medjusobno iskljucenje(“multi exclusion”)
2. Posedovanje i cekanje (“hold and wait”)
3. Kruzno cekanje (“circular wait”) i
4. Nema prekidanja (“no preemption”).

I oni moraju da budu istovremeno ispunjeni.

3. Prikazati kako se vrsi modeliranje potpunog zastoja. Objasniti sta su grafovi dodele resursa I koje tipove cvorova imaju?

Odgovor: Za modeliranje potpunog zastoja koriste se usmereni grafovi. Takvi grafovi se zovu grafovi dodele resursa I imaju dva tipa cvora:-procese koji se prikazuju kao krugovi i
-resurse koji se prikazuju kao kvadrat.

(slike na strain 227).

4. Da li potpuni zastoj obavezno nastaje kada su ispunjena sledeca 4 potrebna uslova:
1. Medjusobno iskljucenje(“multi exclusion”)
2. Posedovanje i cekanje (“hold and wait”)
3. Kruzno cekanje (“circular wait”) i
4. Nema prekidanja (“no preemption”)?

Odgovor: U opstem slucaju moze se reci da:
a) ako u grafu ne postoji kruzni tok nijedan proces nije u zastoju, dok u suprotnom zastoj moze da postoji i
b) ako u grafu postoji kruzni tok I pri tome postoji samo po jedan primerak svakog resursa zastoj postoji.

5. Da li je moguce da nastane potpuni zastoj koji ukljucuje samo jedan proces? Obrazlozi odgovor.

Odgovor: Ne. Zato sto mora postojati bar dva procesa kako bi uopste I doslo do kruznog cekanja.

6. Prikazati primer grafa dodele resursa za sistem kod koga postoji potpuni zastoj?

Odgovor: Slika 9.3. u knjizi strana 228.

7. Prikazati primer grafa dodele resursa za sistem kod koga ne postoji potpuni zastoj?

Odgovor: Slika 9.5. u knjizi, strana 230

8. Dat je sistem koji se sastoji od 5 procesa koji dele 6 resursa istog tipa i gde su svakom procesu potrebna najvise dva resursa. Pokazati da sistem ne moze doci u stanje potpunog zastoja?

Odgovor: Znaci da tri procesa uzimaju sve resurse, a samo su cetvrti i peti procesi u zastoju. Ne postoji kruzno cekanja.(Nisam sigurana koliko je tacan odgovor!!!Pogledati slican primer u knjizi strana 230 i slika 9.5.)

9. Prikazati skup mogucih stanja sistema sa stanovista potpunog zastoja?

Odgovor: skup mogucih stanja sistema sa stanovista potpunog zastoja su: bezbedno, nebezbedno i stanje zastoja. (strana 233)

10. Objasniti strategije sprecavanja i izbegavanja potpunog zastoja?

Odgovor: Stretegija izbegavanja potpunog zastoja zasniva se na cinjenici da se skuo svih stanja sistema moze podeliti u tri podskupa.

11. Cime je odredjeno stanje dodele resursa kod algoritama izbegavanja potpunog zastoja?

Odgovor: Stanje dodele resursa je definisano: - brojem raspolozivih resursa,
- brojem dodeljenih resursa i
- max zahtevima procesa.

12. Navesti primer algoritma za izbegavanje potpunog zastoja?

Odgovor:

13. Objasniti princip rada Bankarovog algoritma?

Odgovor: Za slucaj kada u racunarskom sistemu postoji vise primeraka istog resursa, a u slucaju strategije izbegavanja zastoja neophodno je operativni sistem svaki put, pre dodele nekog resursa, proveri da li sistem posle takve dodele ostaje u bezbednom stanju.

14. Objasniti strategiju koja omogucava otkrivanje potpunog zastoja i nakon toga oporavak sistema?

Odgovor:

15. Da li se za sve sisteme moze usvojiti pretpostavka da se zbog male verovatnoce pojavljivanja potpuni zastoj nikada nece pojaviti?

Odgovor: Ne. U sitemima koji mogu biti kriticni po zivot kao sto je upravljacki system u avionu bezbednost uvek ima prednost u odnosu na jednostavnost sistema. U takvim sistemima neophodna je primena neke od metoda za sprecavanje ili izbegavanje potpunog zastoja.

16. Sistem raspolaze sa 12 primera resursa. Trenutno su aktivna 3 procesa: P1,P2 i P3 sa stanjem prikazanim u sledecoj tabeli:

Proces	Max zahteva	Trenutno dodeljeno
P1	10	5
P2	4	2
P3	9	3

Da li se sistem nalazi u: bezbednom, nebezbednom ili u stanju zastoja?

Odgovor: Sistem se nalazi u nebezbednom stanju. U prikazanom stanju sistema dva primerka resursa su slobodna.Sada jedino process P2 moze da nastavi, ali I kada zavrsi oslobadja cetiri resursa. Tada proces P1 trazi jos pet, a proces P3 jos sest resursa I oba cekaju. Sitem je prema tome u nebezbednom stanju i moze doci do zastoja u slucaju da nijedan proces (P1 i P3)ne oslobodi ranije zauzete resurse.

POGLAVLJE 10 – UPRAVLJANJE PODACIMA

1. Koje su tri najvaznije aktivnosti operativnih sistema u delu za upravljanje podacima na sekundarnoj memoriji?
   1. planiranje dodele sekundarne memorije
   2. dodela slobodne sekundarne memorije
   3. upravljanje slobodnim memorijkim prostorom na sekundarnoj memoriji

2.Navesti funkcije sistema za upravljanje podacima?
   1. identifikovanje i lociranje sekundarne memorije
   2. koriscenje direktorijuma za opisivanje lokacije svih datoteka i njihovih atributa
   3. opis kontrole pristupa korisnika u deljenom sistemu
   4. rad sa blokovima radi pristupa datotekamadodela slobodnih blokova datotekama i
   5. upravljanje slobodnim memorijkim prostorom

3.prikazati graficke komponente sistema za upravljanje podacima
Slika 10.1 strana 240

4.Sa je direktorijum?
Datoteka je lmenovani i postojan skup podataka koji je memorisan na nekom memorijskom medijumu, odnosno na nekom periferijskom uredjaju , kao sto su ”hard ”disk , magnetna traka, ”floppy”disk , ”flash” memorija...Datoteka predstavlja apstrakciju fizickih karakteristika dato memorijskog uredjaja.Podaci u datoteci su postojani cak i po prestanku napajanja.Svaka datoteka ima atribute kao sto su ime, velicina, datum i vreme poslednjeg azuriranja ....
5.Najcesce komande sistema za upravljanje podacima su:
   1. Kreiranje datoteke
   2. citanje i pisanje unutar datoteke radi operacije citanja ili pisanja
   3. pozicioniranje unutar datoteke radi iperacije citanja ili pisanja
   4. postavljanje ili koriscenje mehanizma zastite
   5. promena vlasnistva nad datotekom
   6. listanje datoteka u datom direktorijumu i
   7. brisanje datoteke
   8.

6.Sta je deskriptor datoteke?
Prilikom otvaranja neke datoteke od strane date aplikacije, kernel vraca celobrojnu nenegitvnu vrednost koja se zove deskriptor datoteke i oja na jedinstven nacin identifikuje tu datoteku za sve naredne operacije>

7.Prilikom kreiranja novog procesa kod UNIX operativnog sistema koji se deskriptori datoteka automatski dodeljuju tom novom procesu?
Standarni ulaz (deskriptor 0), standardni izlaz (deskriptor 1), standardni error deskriptor 2).

8.Sta sve moze da prouzrokuje unistenje datoteke ili delova datoteke?
Hardverske greske, otkazi napajanja, otkazi glave diska, prasina, velike temperaturne kao i promene vlaznosti, softverske greske, vandalizam drugih korisnika....

9.Sta je direktorijum?
Direktorijum ili katalog je struktura podataka koja sadrzi listu datoteka i poddirektorijuma.I sam direktorijum je datoteka i omogucava automatsko vodjenje evidencije o datotekama

!Razlika u odnosu na datoteku je u tome sto on sadrzi podatke koji nisu jkorisnicki vec sistemski!

10.Kako se realizuju direktorijumi?
Pomocu sistemskih poziva koji odgovaraju osnovnim operacijama za rad sa datotekama

11.Prkazati linearni prostor imena koji se koristi za rad sa direktorijumima?
Slika 10.2 strana244

12.Prikazati hijerahiski prostor imena koji se koristi za rad sa direktorijumima?
Slika 10.3 Strana 245

13.Prkazati primer aciklicnog grafa ?
Slika 10.4 Strana 246

14.Graficki prikazati primer strukture sistema datoteka?
Slika 10.5 strana 247

15.Koji podaci se nalaze u particijama diska?
Svaka particija ima boot a dalja struktura particije zavisi od diska do diska,Bitan deo particije je i superblok koji sadrzi kljucne parametre o sistemu datoteka kao sto su :
Tip, velicina sistema datoteka, broj slobodnih blokova u sistemu datoteka... pored superbloka tu su i podaci o slobodnim i zauzetim bliokovima datog sistema datoteka u obliku bitmape ili povezane liste pokazivaca,a tu su i direktorijumi i datoteke koje pripedaju datom sistemu datoteka.

16.Tipovi particija?
Primarna , logicka i dodata

17.Navesti organizacije datotka koje se primenjuju kod operatibnih sistema?
Serijska, sekvencijalana, spregnuta, rasuta ili direktna , indeks-sekvancijalna, indeksna sa B-stablima, sa vise kljuceva itd.

18.Primeri sistema datoteka
FAT16, FAT32, NTFS, ext2, ext3 i HPFS.

19.Graficki pokazati strukture podataka koje se koriste kod tipicnog sistema datoteka?
Slika 10.6 strana 251

20.Sta je tabela deskriptora sistema?
Ove tabele sadrze pokazivace na sve datoteke koje je dati proces otvorio

21.Sta je tabela otvoreni datoteka ?Da li se ona kreira za svaki aktivan proces?
U ovoj tabeli se nalaze informacije o pokazivacu na trenutnu poziciju unutar datoteke, o brojacu otvaranja i zatvaranja datoteke i o informacijam potrebnim za lociranje datoteka disku

22.Graficki prikaz strukture podataka koje se koriste za rad sa datotekamakod UNIX operativnog sistema?
slika 10.7 strana 252 (ne verujem da bi nam dali ovako komplikovanu sliku)

23.Osnovne metode za pristup podacima na disku?
sekvencijalni pristup, direktan ili relativni pristup, indeksirani pristup,

24.Objasniti metode pristupa podacima zasnovane na primeni indeksa?
Kod ovakve metode prvo se pretrazuje indeks a zatim se na osnovu pokazivaca direktno pristupa zeljenom slogu.

25.Navesti primer aplikacije koja podacima u datoteci pristupa:
a) sekvencijalno(kod editora i programskih prevodioca)
b) direktno(kod baza podataka)

26.Kako se moze koristiti indeksna datoteka da ubrza pristup datoteci sa direktim pristupom?
Moze se koristiti tako sto su niz blokovaili slogovau datoteci numerisani

27.Kako koriscenje kes memorije moze da poboljsa perfomanse sistema kada se pristupa podacima?Zasto racunarski sistemi ne koriste vise kes memorije kada kes memorija poboljsava perfomanse?
Kes memorija omogucava komponentama racunara da efikasnije komuniciraju privremenim premestanjem podataka sa sporijeg na brzi uredjaj(kes memorja).Visoka cena onemogucava njeno koriscenje u vecoj meri.

28.Zasto se bit mapa mora cuvati na magnetnom mediu perifernog uredjaja, a ne u glavnoj memoriji?

29.Navesti i objasniti osnovne metode dodele prostora na disku?
a) Dodela susednih memorijskih lokacija(dodelom susednih blokova)
b) Dodela povezanih blokova fiksne velicine(susedni blokovi se povezuju u povezanu listu, i koriste se blokovi iste velicine)
c) Koriscenje sema sa indeksima(svaki indeks u tabeli indeksa pokazuje na blokove diska koji sadrze stvarne podatke date datoteke)

30.Objasniti metod dodele susednih blokova.
Jednostavna je za realizaciju, i susednost blokova poboljsava performanse i zbog togfa je omoguceno brzo i jednostavno izracunavanje adrese bloka u kome se nalaze podaci.

31.objasniti metod dodela susednih blokova?
Osnovna prednost ove metode je u tome sto nema eksterne fragementacije , upravljanje memorijom je pojednostavljeno jer su svi blokovi iste velicine

32.Objasniti metod koriscenja sema sa indeksima?
Ovakav nacin omogucuje brz direktan pristup podacima.

33.Cemu sluze udaljeni sistemi datoteka?
Omogucava datom racunaru da”prikljuci”, tj ucini dostupnim jedan ili vise sistema datoteka sa jednog ili vise udaljenih racunara.

34.Sta je LDAP?
Protokol,poznat kao X.509 standard, koji je implementiran u vise operatibnih sistema.Jednostavan protokol za pristup direktorijumuma.Omogucava korisnicima da samo jednim alato za pretrazivanje podataka, pronalaze informacije kao sto su korisnicko ime , sertifikat o bezbednosti...

35.Koji su algoritmi koji se koriste za dodelu diska?
FCFS, SSTF, SCAN, C- SCAN , LOOK

36.Objasniti FCFS?
Najjednostavniji logaritam, izvrsava zahteve za dodelu diska onim redosledom kojim su nastali.

37.Objasniti SSTF?
Algoritam koji u datom trenutku vrsi pokretanje glave diska uzevsi u obzir zadate U/I zahteve.Osnovna ideja je minimiziranje vremena pozicioniranje u odnosu na trenutnu poziciju glave diska>

38.Objasniti SCAN?
Ovaj logaritam razresava problem “gladovanja” koji sse moze javiti kod SSTF algoritma.Problem se razresava tako sto se nakon pokretanja glave diska dalje pokretanje se nastavlja u istom smeru, tj najpe se servisiraju zahtevi pocev od najdalje spoljasnje staze ka unutrasnjoj stazi koja je krajnja, a zatim od najblize unutrasnje ka najdaljoj spoljasnjoj.

39.Objasniti C-SCAN?
Ovaj algoritam omogucuje uniformnija vremena odziva tako sto se pokratanje glava diska uvek vrsi u jednom smeru.

40.Objasniti LOOK?
Ovaj algoritam modifikuje SCAN algoritam, tako sto zaustavlja kretanje glave u istom smeru ako nema vise U/I zahteva u tom smeru.

41.Objasniti C-LOOK?
Ovaj algoritam servisira zahteve pomeranjem glave diska od spoljasnjeg ka unutrasnjem cilindru dokle god ima zahteva, a zatim se vraca na spoljasnji najudaljeniji cilindar za koji postoji zahtev.

POGLAVLJE 11: DISTRIBUIRANI SISTEMI

1) Koje su osnovne karakteristike distribuiranih sistema?

- Osnovne karakteristike distribuiranih sistema su deljenje resursa, transparentnost, otvorenost i uravnotezenost.

2) Objasniti sta je transparentnost.

- Transparentnost je osobina sistema da se od korisnika sakrije cinjenica da su hardverske i softverske komponente distribuirane po razlicitim racunarskim sistemima. Korisnik vidi sistem kao celinu, a ne kao skup nezavisnih komponenti.

3) Objasniti sta je transparentnost pristupa.

- Transparentnost pristupa se odnosi na sakrivanje nacina reprezentacije podataka i nacina pristupa resursima u sistemu.Time se omogucava pristup lokalnim i udaljenim (distribuiranim) resursima na uniforman nacin.

4) Objasniti sta je transparentnost lokacije.

- Transparentnost lokacije se odnosi na sakrivanje lokacije resursa u distribuiranom sistemu. Korisnik ne mora da zna lokaciju resursa da bi mu pristupio.

5) Objasniti sta je transparentnost migracije.

- Transparentnost migracije se odnosi na cinjenicu da se resursi iz razlicitih razloga najcesce zbog poboljsanja performansi, mogu premestiti sa jednog racunara na drugi racunar u mrezi.

6) Objasniti sta je transparentnost relokacije.

- Transparentnost relokacije se odnosi na situaciju kada se resurs premesta sa jedne lokacije na drugu u toku pristupa, odnosno u toku koriscenja resursa.

7) Objasniti sta je transparentnost replikacije.

- Transparentnost replikacije znaci da postojanje vise kopija (replika) jednog resursa rasporedjenih na razlicite lokacije u mrezi treba da bude sakriveno od korisnika i korisnickih aplikacija.

8) Objasniti sta je transparentnost konkurencije.

- Transparentnost konkurencije se odnosi na situaciju kada vise procesa istovremeno koristi neki deljivi resurs.

9) Objasniti sta je transparentnost otkazivanja.

- Transparentnost otkazivanja oznacava da otkaz neke hardverske ili softverske komponente u distribuiranom sistemu ne sme da omete izvrsavanje zahteva korisnika.

10) Objasniti sta je transparentnost persistencije.

- Transparentnost persistencije se odnosi na skrivanje cinjenice da je zahtevani resurs u memoriji ili na disku.

11) Objasniti sta su otvoreni sistemi.

- Otvoreni sistemi podrzavaju servise za koje je definisana sintaksa i semantika servisa. Takvi servisi su specificirani posredstovm pogodnih jezika za specifikaciju interfejsa IDL (“Interface Definition Language”).

12) Objasniti sta je uravnotezenost ili srazmernost sistema("scalability").

- Za sistem se kaze da je uravnotezen kada njegove performanse ne opadaju pri sirenju sistema po velicini (broju korisnika i broju resursa). Tj ako jedan server opsluzuje 20 korisnika, tada dva servera treba da opsluze 40 korisnika.

13) Objasniti heterogenost distribuiranih sistema.

- U distribuiranim sistemima heterogenost se javlja na prakticno svim nivoima, kao sto su racunarske mreze, hardver racunara, operativni sistemi, programski jezici i razlicite implementacije pojedinih komponenti.

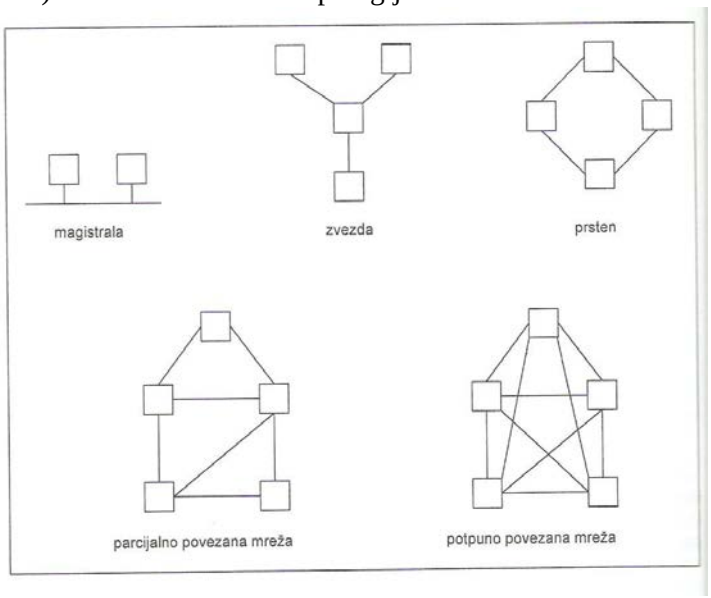
14) Objasniti raspolozivost resursa i pouzdanost podataka u distribuiranim sistemima.

- Raspolozivost resursa znaci da i pored otkaza pojedinih resursa u sistemu, postoje resursi koji su i dalje dostupni i da se posao moze zavrsiti uspesno. Tj ako neki resurs ili funkcionalna jedinica resursa na jednoj lokaciji u mrezi otkaze zahtevana obrada se moze izvrsiti na resursima koji se nalaze na drugim lokacijama u mrezi. Pouzdanost podataka cesto je povezana sa replikacijom podataka u distribuiranom sistemu.

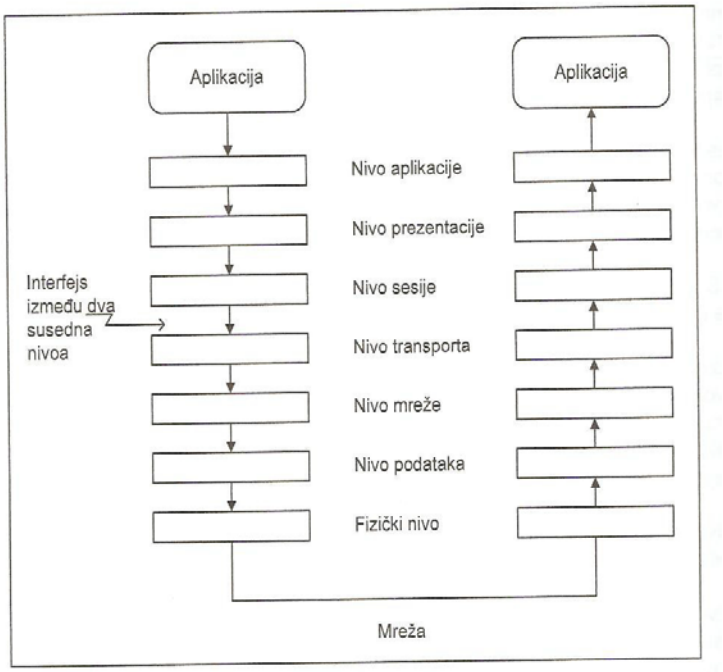
15) Navesti primere distribuiranih sistema.

- Najpoznatiji primer distribuiranih sistema je svakako Internet, ali postoji i intranet.

16) Prikazati osnovne topologije racunarskih mreza.



17) Prikazati OSI referentni model za komunikacije.

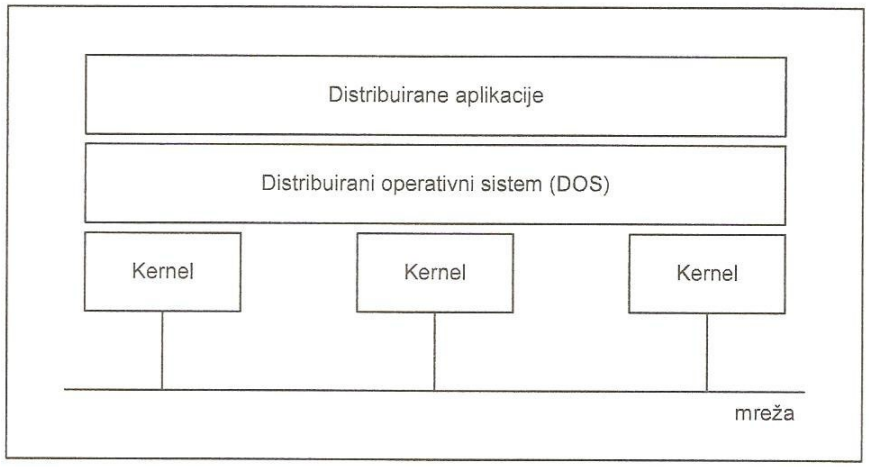


ili moze tekstom:
OSI model se sastoji od sledecih funkcija: 1) Fizicki nivo 2) Nivo podataka 3) Nivo mreze 4) Nivo transporta 5) Nivo sesije 6) Nivo prezentacije 7) Nivo aplikacije

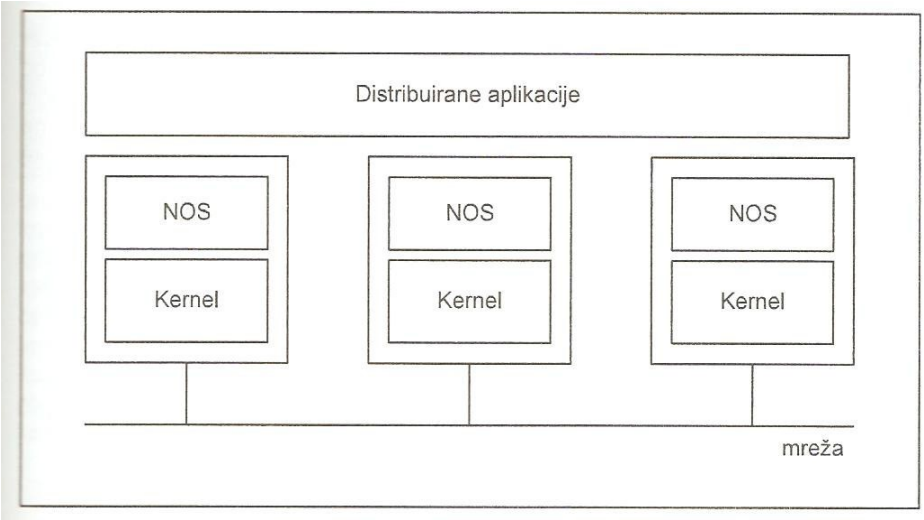
18) Objasniti ukratko funkcije pojedinih nivoa OSI modela.
- Fizicki nivo je odgovoran za prenos niza bitova. Jedan od najpoznatiijh protokola ovog nivoa je RS-232. Nivo podataka obezbedjuje otkrivanje i ispravljanje gresaka do kojih moze doci prilikom prenosa. Nivo mreze je odgovoran za obezbedjivanje konekcije izmedju dva cvora, najpoznatiji protokoli su X.25 i IP. Nivo transporta obezbedjuje ispravan prenos poruke kroz mrezu, i najpoznatiji protokol je TCP. Nivo sesije kontrolise dijalog izmedju dva cvora u mrezi. Nivo prezentacije omogucuje promenu nacina reprezenotvanja poruke (ukoliko postoje razlike u predstavljanju brojeva u pokretnom zarezu,ukoliko postoji enkripcija ili neka kompresija). Nivo aplikacije podrzava servise prenosa podataka, elektronsku postu i druge. Najpoznatiji protokol ovog nivoa je FTP.

19) Objasniti protokol za povezivanje metode udaljenih objekata.
- Protokol za povezivanje metode udaljenih objekata ili skraceno RMI (“Remote Method Invocation”), cine tri osnovne operacije doOperation, getRequest i sendReply. Klijent izvrsava operaciju doOperation kojom zahteva izvrsenje odredjene metode. Operaciju getRequest obavlja server proces. Po izvrsenju specificirane metode server obavlja operaciju sendReply kojom salje odgovor klijentu.

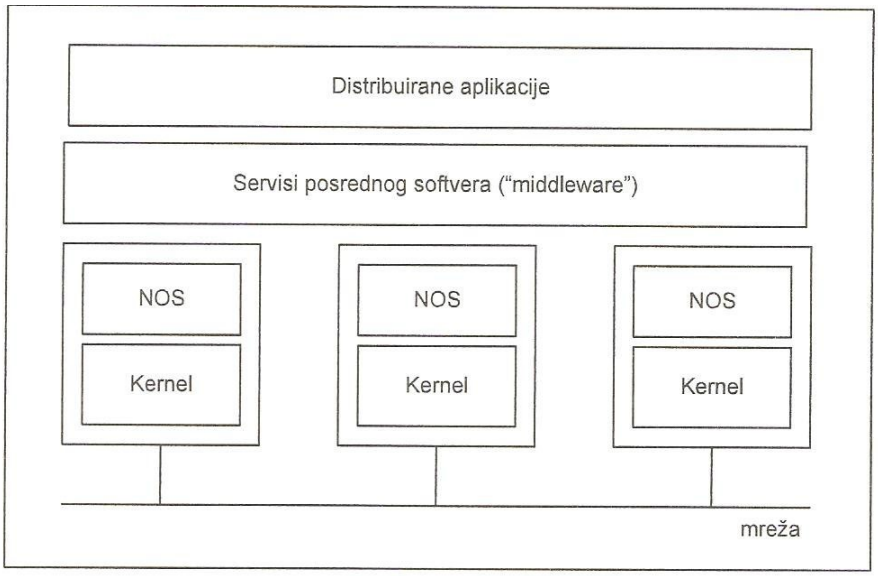
20) Prikazati arhitekturu distribuiranih operativnih sistema.



21) Prikazati arhitekturu mreznih operativnih sistema.



22) Prikazati polozaj posrednog softvera (“middleware”) u distribuiranim sistemima.

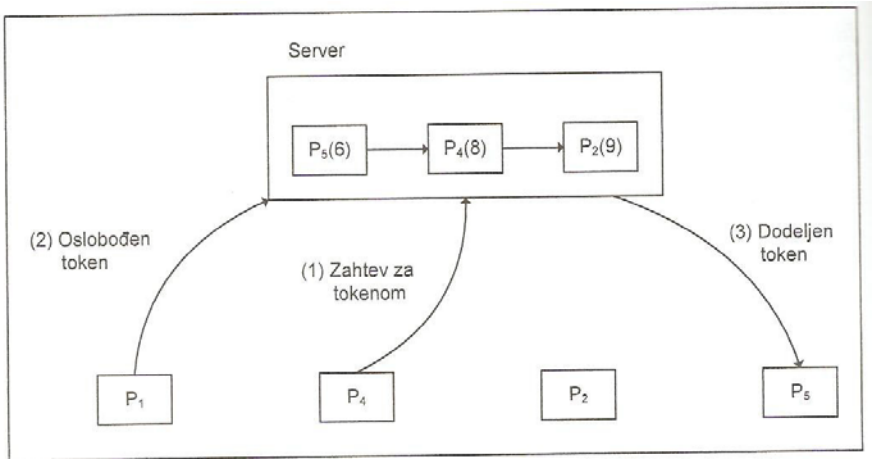


23) Kako se vrsi sinhronizacija logickih satova u distribuiranim sistemima?
- Sinhronizacija se vrsi po UTC vremenu ili Lamportovim algoritmom.

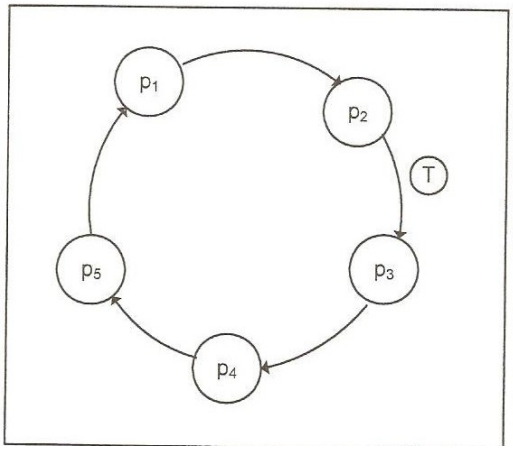
24) Objasniti sta je globalno stanje u distribuiranim sistemima?
- Globalno stanje u distribuiranom sistemu definisano je lokalnim stanjem svakog procesa i poruka koje su u fazi prenosa (poruka koje su poslate ali nisu stigle na odrediste).

25) Objasniti distribuirano medjusobno iskljucenje procesa.
-

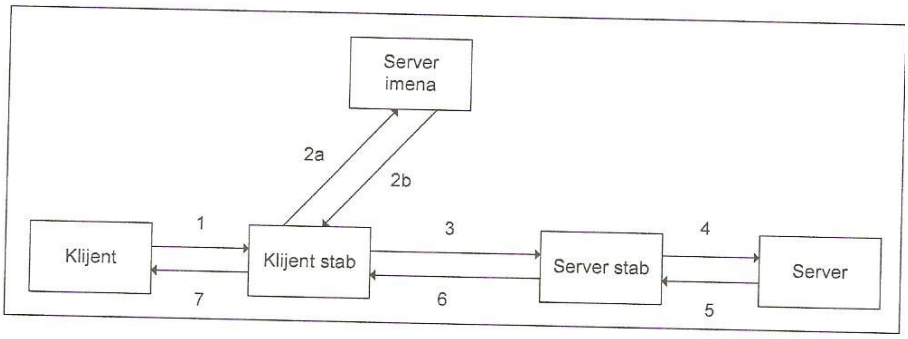
26) Prikazati realizaciju algoritma sa tokenima za distribuirano medjusobno iskljucenje procesa.



27) Prikazati realizaciju algoritma za distribuirano medjusobno iskljucenje procesa gde ne postoji koordinator.



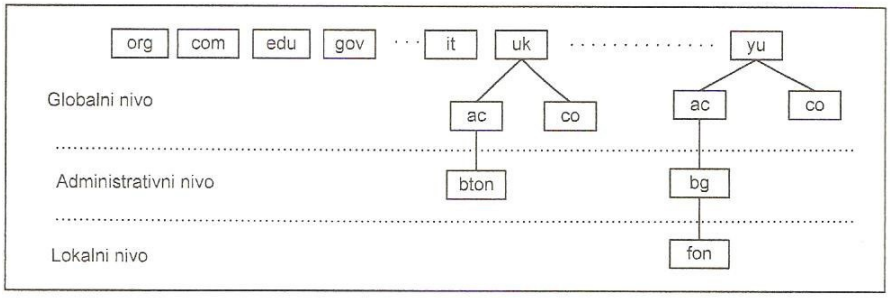
28) Prikazati ulogu staba ("client and server stub") i koncept realizacije poziva udaljenih procedura.



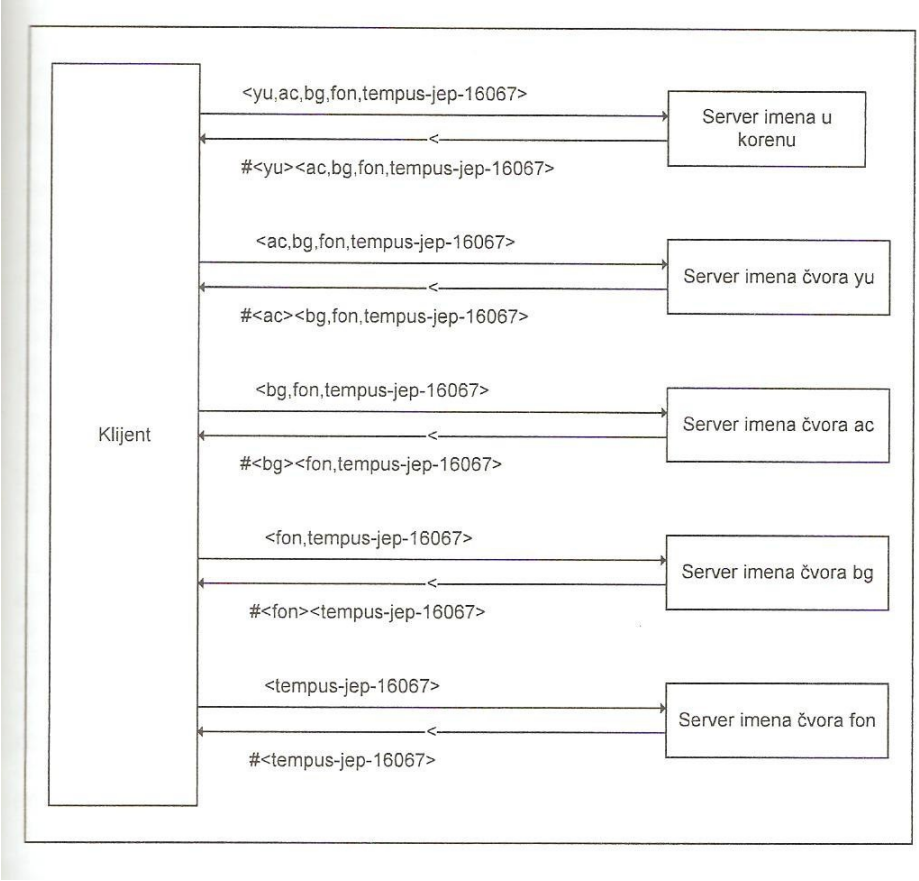
29) Objasniti imenovanje kao jedan od najvaznijih servisa koje posredni softver ("middleware") podrzava.

- Pod imenovanjem se podrazumeva dodeljivanje imena komponentama, resursima, servisima, u skladu sa usvojenim sistemom imenovanja i razresavanja imena, odnosno preslikavanje imena u adresu lokacije na kojoj se entitet nalazi.

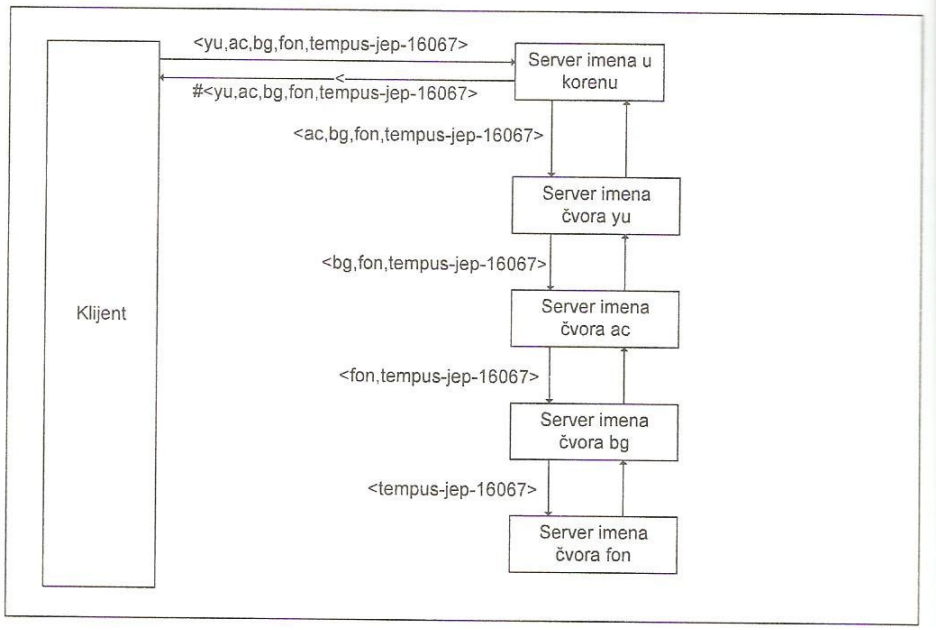
30) Prikazati primer jednog dela prostora imena DNS ("Domain Name Space").



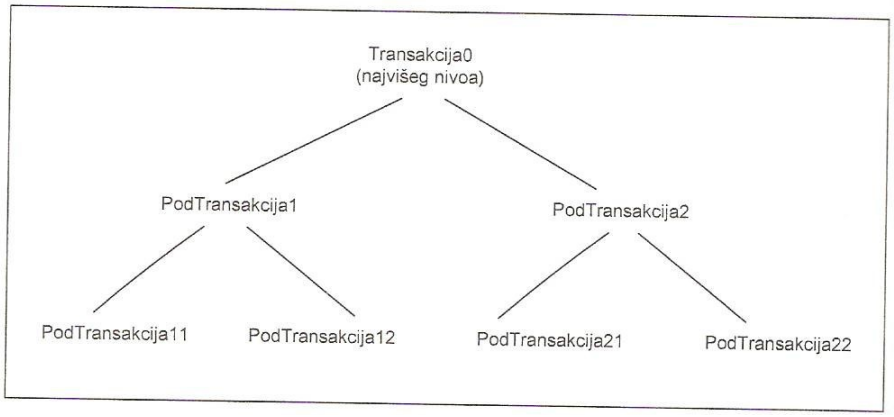
31) Prikazati i objasniti nacin razresavanja imena zasnovan na iterativnom postupku.



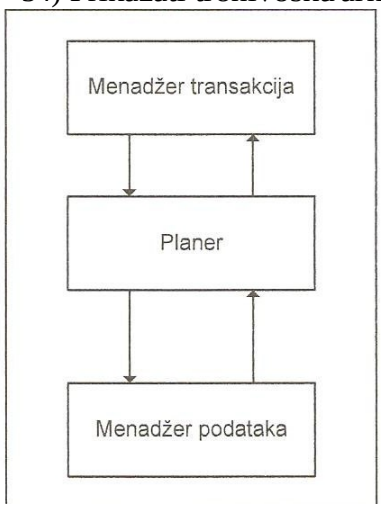
32) Prikazati i objasniti rekurzivni nacin razresavanja imena.



33) Prikazati i objasniti sta su ugnjezdene transakcije.



34) Prikazati tronivosku arhitekturu softvera za upravljanje konkurentim transakcijama.



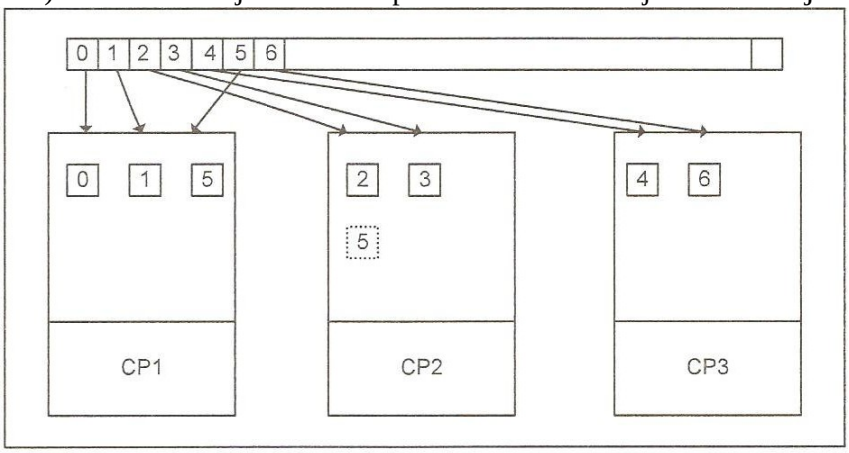
35) Objasniti sta je replikacija.

- Replikacija znaci umnozvanje resursa u sistemu, radi povecanja pouzdanosti i poboljsanja perormansi sistema.

36) Navesti znacajna pitanja u implementaciji replikacije.

- Znacajno pitanje u implementaciji replikacije je to ko kreira repliku, gde je i kada je kreira.

37) Prikazati i objasniti koncept distribuirane deljene memorije.



38) Definisati pojam distribuiranog sistema datoteka.
- Distribuirani sistem datoteka predstavlja skup datoteka koje se nalaze na razlicitim racunraskim sistemima u racunarskoj mrezi. DSD omogucava korisnickim procesima pristup i rad sa datotekama koje su uskladistene na lokalnim diskovima udaljenih racunara.

39) Objasniti sta je semantika deljenja datoteka. Navesti i objasniti primere,
- Semantika deljenja definise kako ce se akcije jednog programa na jednoj datoteci videti od strane drugih programa koji konkurentno koriste tu istu datoteku. Primeri su Unix semantika, semantika seanse, semantika nepromenljivih deljenih datoteka, semantika tipa transakcija

40) Objasniti metode pristupa.
- Model udaljenog pristupa se desava kada datoteta kojoj se pristupa se nalazi na udaljenom serveru koji poseduje svoj sistem za upravljanje podacima. Klijent preko interfejsa ispostavlja zahtev za obradom koje izvrsava server, a zatim rezultate obrade salje klijentu. Drugi nacin je lokalna obrada podataka, kada klijent obdajuje lokalno datoteku tako sto je dobija (download) od servera, i nakon obrade salje je serveru (upload) da se moze koristiti od strane drugih klijenata. Koristi se kod FTP servisa interneta.

41) Objasniti i uporediti pojmove transparentnosti lokacije i nezavistnosti lokacije.
- Transparentnost lokacije znaci da ime datoteke ne treba da otkrije fizicku lokaciju datoteke u distribuiranom sistemu, dok kod nezavnisnosti lokacije sistem moze da promeni fizicku lokaciju datoteke i da pri tome ne mora da promeni ime datoteke.

42) Objasniti razliku izmedju servera sa ocuvanjem stanja i servera bez ocuvanja stanja.
- Server sa ocuvanjem stanja kako mu i samo ime kaze cuva stanje datoteke (od trenutka otvaranja datoteke) i smesta ga u AFT tabelu. Server bez ocuvanja stanja, ne cuva ove podatke vec to rade klijenti.

43) Uporediti performanse i otpornost na otkaze izmedju servera sa ocuvanjem stanja i servera bez ocuvanja stanja.
- Ukoliko server bez ocuvanja stanja otkaze informacije o obradi se ne gube jer klijent cuva stanje obrade. Ukoliko server sa ocuvanjem stanja otkaze gube se svi podaci o obradi i obustavlja se rad.

44) Objasniti razliku izmedju sposobnosti za oporavak i robustnosti datoteke.
- Kaze se za datoteku da poseduje sposobnosti za oporavak ukoliko po otkazu jedne operacije klijenta ili prestanka rada klijenta, je moguce da se povrati u prethodno konzistentno stanje. Robustne datoteke garantuju da mogu da opstanu posle otkaza ili kvara medijuma za skladistenje.

45) Objasniti razliku izmedju sposobnosti za oporavak i raspolozivosti datoteke.
- Kaze se za datoteku da poseduje sposobnosti za oporavak ukoliko po otkazu jedne operacije klijenta ili prestanka rada klijenta, je moguce da se povrati u prethodno konzistentno stanje. Za datoteku se kaze da je raspoloziva kada joj se moze pristupiti uprkoz otkazu servera, uredjaja za skladistenje ili komunikacija (postize se replikacijom).