Lekcija 1

1. **Koje poslove obavlja operativni sistem?**

Operativni sistem **upravlja sastavnim delovima računara** (procesorom, kontrolerima i random memorijom) i **stvara pristupačno radno okruženje za korisnika** (pretvara računar u mašinu koja rukuje datotekama i procesima) /**upravlja fizičkim i logičkim delovima računara**/

2. **Šta obuhvata pojam datoteke?**

Pojam datoteke obuhvata **sadržaj i atribute** datoteke. (sadržaj=korisnički podaci, atributi=veličina/vreme nastanka)

3. **Šta se nalazi u deskriptoru datoteke?**

U deskriptoru datoteke nalaze se **atributi**.

4. **Šta omogućuju datoteke?**

Datoteke omogućuju **trajno čuvanje podataka**.

5. **Šta obavezno prethodi čitanju i pisanju datoteke?**

Čitanju i pisanju prethodi **otvaranje datoteke**. Time se omogućuje priprema za pristup podacima (deskriptor se prebacuje sa masovne u radnu memoriju).

6. **Šta sledi iza čitanja i pisanja datoteke?**

Iza čitanja i pisanja sledi **zatvaranje datoteke**. Time se sačuvaju atributi i sadržaj (prebacuje se deskriptor iz radne u masovnu memoriju).

7. **Šta obuhvata pojam procesa?**

Pojam procesa obuhvata **aktivnost, sliku i atribute** procesa. (aktivnost=angažovanje na izvršavanju korisničkog programa, slika=adresni prostor procesa sa naredbama izvršavanja programa, stekom i podacima koji se obrađuju, atributi=stanje i prioritet)

8. **Šta se nalazi u deskriptoru procesa?**

U deskriptoru se nalaze **atributi procesa (stanje i prioritet)**.

9. **Koja stanja procesa postoje?**

Tipična stanja su **aktivan, čeka i spreman**. (aktivan=izvršava program, čeka=nisu ispunjeni preduslovi za obradu podataka, spreman=samo zauzetost procesora onemogućuje izvršavanje programa)

10. **Kada je proces aktivan?**

Proces je aktivan **kada procesor izvršava program**.

11. Šta je kvantum?

Kada postoji više procesa istog prioriteta, bitna je raspodela procesorskog vremena između njih. **Aktivan proces prepušta procesor spremnom procesu** najvišeg prioriteta **čim istekne unapred određen vremenski interval** koji se naziva **kvantum**.

12. Šta se dešava nakon isticanja kvantuma?

Aktivan proces prepušta procesor spremnom procesu najvišeg prioriteta.

13. Po kom kriterijumu se uvek bira aktivan proces?

Od prioriteta procesa zavisi kada će spreman da postane aktivan proces i podrazumeva se da je aktivan uvek onaj sa najvišim prioritetom.

14. Koji prelazi su mogući između stanja procesa?

čeka u spreman, spreman u aktivan, aktivan u čeka, aktivan u spreman

15. Koji prelazi nisu mogući između stanja procesa?

spreman u čeka i čeka u aktivan

16. Šta omogućuju procesi?

Procesi omogućuju **bolje iskorišćenje procesora i njegovu bržu reakciju** na spoljašnje događaje. (postojanje više spremnih procesa omogućuje stalno iskorišćenje procesa, a brža reakcija omogućena je tako što je manje prioritetan proces aktivan dok prioritetniji proces čeka neki spoljni događaj)

17. Šta karakteriše sekvencijalni proces?

Sekvencijalni proces karakteriše **trag** (redosled izvršavanja naredbi programa) **koji je određen u vreme programiranja i zavisi samo od obrađivanih podataka**.

18. Šta karakteriše konkurentni proces?

Konkurentni procesi su procesi sa **više istovremeno postojećih niti**. Niti se međusobno prepliću u redosledu koji nije određen u vreme programiranja.

19. Šta ima svaka nit konkurentnog procesa?

Svaka nit ima svoj **prioritet, stanje** i **stek**, pa i svoj **deskriptor**. (niti nisu potpuno samostalne, međusobno sarađuju razmenom podataka)

20. Koju operaciju uvodi modul za rukovanje procesorom?

Jezgro operativnog sistema (kernel) se može raščlaniti na module za rukovanje procesorom, kontrolerima, random memorijom, datotekama i procesima. Modul za rukovanje procesorom uvodi **operaciju preključivanja**.

21. Po čemu se razlikuju preključivanja između niti istog procesa i preključivanja između niti raznih procesa?

S obzirom da su **niti istog procesa** u istom adresnom prostoru, a niti različitih procesa u različitim adresnim prostorima, pri preključivanju u prvom slučaju **ne dolazi do promene adresnog prostora, pa se preključivanje brže obavlja**.

22. Koje operacije uvodi modul za rukovanje kontrolerima?

Drajveri modula za rukovanje kontrolerima uvode **operacije ulaza i izlaza**.

23. Šta karakteriše drajvere?

Svaki drajver je **specijalizovan za jednu vrstu kontrolera,** opslužuje jednu klasu ulazno-izlaznih uređaja i ima zadatak da konkretan ulazno-izlazni uređaj predstavi u apstraktnom obliku sa jednoznačnim i pravilnim načinom korišćenja. Obrađivači prekida takođe ulaze u sastav drajvera.

24. Koje operacije uvodi modul za rukovanje radnom memorijom?

Modul za rukovanje radnom memorijom uvodi **operacije zauzimanja i oslobađanja**, koje dovode do zauzimanja i oslobađanja memorije.

25. Koje operacije poziva modul za rukovanje radnom memorijom kada podržava virtuelnu memoriju?

Poziva **operacije ulaza i izlaza**. Kada podržava virtuelnu memoriju pored evidencije o slobodnoj radnoj memoriji radi zauzimanja i zauzetoj radi oslobađanja, brine i o prebacivanju sadržaja stranica između radne i masovne memorije kada poziva pomenute operacije.

26. Koje operacije uvodi modul za rukovanje datotekama?

Modul za rukovanje datotekama uvodi **operacije otvaranja, zatvaranja, čitanja i pisanja**. (čitanje i pisanje podrazumeva prenos sadržaja datoteke između radne i masovne memorije)

27. Koje operacije poziva modul za rukovanje datotekama?

S obzirom da modul za rukovanje datotekama pored otvaranja, zatvaranja, čitanja i pisanja vodi evidenciju o memorijskim blokovima masovne memorije u kojima je sadržaj datoteke, te o prebacivanju tog sadržaja između radne i masovne memorije ovaj modul poziva **operacije ulaza i izlaza**. Pošto su potrebni veliki baferski prostori za prebacivanje pomenutog sadržaja, ovaj modul poziva **i operaciju zauzimanja**.

28. Šta omogućuje *multiprocessing* i *multithreading*?

Multiprocessing je **višeprocesni režim rada** (mogućnost istovremenog postojanja više procesa), a multithreading je **mogućnost istovremenog postojanja više niti.**

29. Koje operacije uvodi modul za rukovanje procesima?

Modul za upravljanje procesima uvodi **operacije stvaranja i uništavanja** (stvara se ili uništava proces ili nit).

30. Koje operacije poziva modul za upravljanje procesima?

Modul za upravljanje procesima poziva **operaciju čitanja** radi preuzimanja sadržaja datoteka potrebnih za stvaranje slike procesa. Takođe poziva **operacije zauzimanja i oslobađanja** jer mu je za stvaranje slike procesa neophodna radna memorija.

31. Koje module sadrži slojeviti operativni sistem?

Od vrha hijerarhije: **modul za rukovanje procesima, modul za rukovanje datotekama, modul za rukovanje random memorijom, modul za rukovanje kontrolerima, modul za rukovanje procesorom.** Pravilo: Iz svakog sloja se pozivaju samo operacije uvedene u nižim slojevima. Monolitni operativni sistemi - bez hijerarhijske strukture.

32. Šta omogućuju sistemski pozivi?

Sistemski pozivi omogućuju **prelazak iz korisničkog u sistemski prostor** radi poziva operacija operativnog sistema.

33. Koje adresne prostore podržava operativni sistem?

Logički (virtuelne adrese) i **fizički** (adrese memorije) adresni prostor.

34. Šta karakteriše interpreter komandnog jezika?

IKJ (shell) je proces iz korisničkog sloja koji **preuzima i interpretira komande komandnog jezika** koje precizno opisuju rukovanje procesima i datotekama čime **lišavaju korisnika potrebe za poznavanjem mehanizama** unutar operativnog sistema. IKJ posreduje između operativnog sistema i korisnika.

35. Koji nivoi korišćenja operativnog sistema postoje?

IKJ koristi operativni sistem na **programskom nivou-povlašćeni** (poziva sistemske operacije), a korisnik na **korisničkom nivou-interaktivni**.

Lekcija 2

1. **Šta je preplitanje?**

Mešanje izvršavanja naredbi raznih niti, odnosno niti i obrađivača prekida se naziva preplitanje.

2. **Da li preplitanje ima slučajan karakter?**

Preplitanje ima slučajan karakter, jer nije u napred poznato posle koje naredbe će se desiti prekid i eventualno preključivanje. To nije moguće odrediti čak ni za prekide pravilnog perioda (prekide sata).

3. **Šta izaziva pojavu preplitanja?**

Činjenica da nije u napred poznato posle koje naredbe će se desiti prekid i eventualno **preključivanje**. (Procesor izvršava jednu nit, desi se prekid koji izazove preključivanje i procesor u nastavku izvršava drugu nit.)

4. **Da li preplitanje može uticati na rezultat izvršavanja programa?**

Pod uticajem preplitanja **rezultati mogu biti stohastični**, odnosno mogu da se menjaju od izvršavanja do izvršavanja.

5. **Šta su deljene promenljive?**

To su **promenljive kojima pristupaju niti i obrade prekida ili razne niti**, te se može reći da ih međusobno dele. (Klase za rukovanje deljenim promenljivama su deljene klase).

6. **Šta je preduslov očuvanja konzistentnosti deljenih promenljivih?**

Da se rukovanje deljenim promenljivama obavlja sekvencijalno. (Novo izvršavanje bilo koje od operacija deljene promenljive može početi tek nakon završetka prethodno započetog izvršavanja neke od operacija deljene promenljive, tako svako od izvršavanja ostavlja i zatiče deljene promenljive u konzistentnom/predviđenom stanju). Ovo ne važi kod štetnih preplitanja. Da bi se sprečila štetna preplitanja mora se obezbediti međusobna isključivost.

7. **Šta su kritične sekcije?**

Tela operacija deljenih klasa ili delovi ovih tela čije izvršavanje je kritično za konzistentnost deljenih promenljivih.

8. **Šta je sinhronizacija?**

Proces vremenskog usklađivanja izvršavanja kritičnih sekcija je sinhronizacija, time se postiže njihova medjusobna isključivost.

9. Koje vrste sinhronizacije postoje?

Sinhronizacija **za ostvarenje međusobne isključivosti i uslovna sinhronizacija** (ona na primer sprečava pražnjenje nenapunjenog bafera odnosno punjenje napunjenog bafera). Zasniva se na zaustavljanju aktivnosti niti dok se ne ispuni uslov neophodan za nastavak aktivnosti.

10. Šta je atomski region?

Kritične sekcije u kojima su onemogućeni prekidi, pa samim tim ne dolazi ni do obrada prekida koje mogu dovesti do štetnog preplitanja. Atomski regioni imaju neprekidnost izvršavanja odnosno **nedeljivi su**, što garantuje njihovu međusobnu isključivost.

11. Šta sužava primenu atomskih regiona?

Važnost da izvršavanja atomskih regiona budu što kraća, jer onemogućenje prekida u atomskim regionima odlaže obradu novih prekida i usporava reakciju procesora na spoljne događaje.

12. Čemu služi propusnica?

Svaka deljena promenljiva poseduje propusnicu za ulazak u njene kritične sekcije da bi se na taj način omogućila **međusobna isključivost kritičnih sekcija zasnovana na zaustavljanju aktivnosti niti kada ona pokuša da uđe u kritičnu sekciju deljene promenljive** kojoj već pristupa neka druga nit (a ne zasniva se na onemogućenju prekida). Pripusnica je slobodna ili zauzeta, a bez nje se ne može ući u kritičnu sekciju.

13. Šta se dešava sa niti koja zatraži, a ne dobije propusnicu?

Propusnicu traže sve niti koje se takmiče za ulazak u kritičnu sekciju deljene promenljive, jedna dobija propusnicu, a ostale **zaustavljaju svoju aktivnost i prelaze u stanje *čeka* i ostaju u njemu do eventualnog dobijanja trežene propusnice.**

14. Šta se dešava kada nit vrati propusnicu?

Tada **jedna od niti koja je u stanju *čeka* dobije propusnicu i prelazi u stanje *spremna,* a u kritičnu sekciju ulazi kada postane aktivna** tj. kada se procesor preključi na nju.

15. Kako se štiti konzistentnost propusnica?

S obzirom da su rukovanja propusnicom kratkotrajna konzistentnost propusnica se štiti **korišćenjem atomskih regiona** (tada rukovanje propusnicom nije ugroženo štetnim preplitanjima).

16. Šta je isključivi region?

Kritične sekcije koje međusobnu isključivost ostvaruju korišćenjem propusnica (razlikuju se od atomskih regiona).

17. Šta uvode poželjne osobine konkurentnih programa?

Svaka od poželjnih osobina uvodi **ili tvrdnju isključivanja nepoželjnog ili tvrdnju uključivanja poželjnog** (safety i liveness property). Primeri safety property: u izvršavanjima ne nastaju pogrešni rezultati, u izvršavanjima nema narušavanja međusobne isključivosti kritičnih sekcija. Primeri liveness property: izvršavanje ima kraj, dese se svi zatraženi ulasci u kritične sekcije u toku izvršavanja programa.

18. Po čemu se konkurentno programiranje razlikuje od sekvencijalnog?

(Zbog slučajne prirode preplitanja, tačan rezultat dobijen u jednom ili više izvršavanja ne isključuje mogućnost postojanja izvršavanja koja za iste ulazne podatke daju netačan rezultat – to nije način provere ispravnosti konkurentnog programa.) **Razlikuju se po rukovanju nitima i deljenim promenljivama.**

19. Koje prednosti ima konkurentna biblioteka u odnosu na konkurentni programski jezik?

Kod nastanka konkurentnog programskog jezika neizbežne su **aktivnosti vezane za definisanje sintakse i semantike programskog jezika i aktivnosti vezane za zahvate na kompajleru**. Te aktivnosti **se izbegavaju zasnivanjem konkurentnog programiranja na korišćenju konkurentne biblioteke**. Prednost korišćenja biblioteke je što **omogućuje korišćenje postojećeg poznatog programskog jezika**.

20. Kako se opisuju niti?

Rukovanje nitima opisuje klasa *thread*. Njen konstruktor kreira nit. **Kao argumet poziva konstruktora se navodi adresa funkcije koja opisuje nit**. Šta nit treba da radi napisano je u telu funkcije, a kraj niti je predviđen kada se završi telo te funkcije.

21. Kako se kreiraju niti?

Rukovanje nitima opisuje **klasa *thread*. Njen konstruktor kreira nit**. Operacija *join()* klase *thread* zaustavlja aktivnost svog pozivaoca dok se ne završi aktivnost niti na koju se operacija odnosi. Operacija *detach( )* saopštava da regularan kraj niti na koju se operacija odnosi može da nastupi kao posledica kraja aktivnosti procesa kome ta nit pripada.

22. Kada se zauzima propusnica deljene promenljive?

Pre obavljanja operacija (izlaznih i ulaznih) nad deljenom promenljivom (**pre ulaska u kritičnu sekciju**).

23. Kada se oslobađa propusnica deljene promenljive?

Nakon završetka obavljanja operacija nad deljenom promenljivom (**po izlasku iz kritične sekcije**).

24. Kakvu ulogu ima klasa *mutex*?

Klasa *mutex* **omogućuje međusobnu isključivost kritičnih sekcija**. **Njen objekat predstavlja propusnicu.**

25. Kakve operacije sadrži klasa *mutex*?

Operacije ***lock( )*** za zauzimanje propusnice **i *unlock( )*** za njeno oslobađanje.

26. Kakvu ulogu ima klasa *unique_lock*?

Korišćenjem templejt klase *unique_lock* **postiže se** takođe **međusobna isključivost kritičnih sekcija**.

27. Kakve operacije sadrži klasa *unique_lock*?

Njen **konstruktor poziva operaciju *lock( )*, a njen destruktor operaciju *unlock( )* klase *mutex*** i to tako što se kao argument klase *unique_lock* navodi klasa *mutex*, a kao argument njenog konstruktora objekat klase *mutex*.

28. Kakvu ulogu ima klasa *condition_variable*?

Za ostvarenje uslovne sinhronizacije. Ako uslov neophodan za dalju aktivnost niti nije ispunjen ona se zaustavlja.

29. Kakve operacije sadrži klasa *condition_variable*?

Sadrži operacije *wait( ) i notify_one( ).* Ove operacije se koriste u isključivom regionu. Operacija *wait( )* omogućava zaustavljanje aktivnosti niti koja ju je pozvala do ispunjenja uslova za njenu dalju aktivnost. Pri zaustavljanju niti ona se prevodi u stanje *čeka*, oslobađa se propusnica prosleđena operaciji kao argument (objekat klase *unique_lock)* i procesor se preključuje na drugu nit koja je spremna.

30. U pozivu koje operacije klase *condition_variable* se vraća propusnica?

Operacija *notify_one( )* omogućuje objavu ispunjenja uslova kako bi nit koja čeka na to nastavila sa svojom aktivnošću. Nit koja je pozvala tu operaciju tada oslobađa propusnicu koju preuzima obaveštena nit.

31. Koje vrste razmene poruka postoje?

Sinhrona i asinhrona razmena.

32. U čemu se razlikuju sinhrona i asinhrona poruka?

Kod asinhrone razmene poruka se aktivnost pošiljaoca zaustavlja samo kada je komunikacioni kanal kroz koji šalje poruke pun, a aktivnost primaoca samo kada je kanal prazan. Ako komunikacioni kanal ima kapacitet više poruka onda svakom prijemu može prethoditi više slanja. Ako je kapacitet kanala jedna poruka, razmena je sinhrona. Kod sinhrone razmene se uvek zaustavlja aktivnost niti koja prva započne razmenu (primalac ili pošiljalac) dok i druga ne započne razmenu poruka. Kod sinhrone razmene nema upisa poruke u komunikacioni kanal. Sinhrona razmena se naziva randevu jer zahteva da se pošiljalac i primalac poruke sretnu. **Dakle, razlikuju se u tome što kod sinhrone razmene poruka niti primalac i pošiljalac direktno razmenjuju poruke** (čija se adresa upisuje u pregradak koji se anulira kada je prazan, odnosno kad postoji poruka obaveštava se primalac da je preuzme pa se pregradak anulira za smeštanje nove poruke) **i čekaju jedna na drugu, dok kod asinhrone razmene se podaci upisuju u nekog posrednika i one se zaustavljaju samo u slučaju da je posrednik pun/prazan.** Kod sinhrone razmene mora biti obezbeđena međusobna isključivost pristupa lokalnoj promenljivoj pošiljaoca u kojoj je poruka.

33. Šta omogućuje funkcija *sleep_for( )*?

Omogućuje uspavljivanje niti. Uspavljuje se nit, pozivalac funkcije, i aktivira se najprioritetnija spremna nit. Buđenje niti je najranije nakon perioda zadatog u argumentu funkcije u milisekundama.

34. Po kojim ciljevima se konkurentno programiranje razlikuje od sekvencijalnog programiranja?

Jedini cilj sekvencijalnog programiranja je *opisivanje obrada podataka*. To je ujedno i osnovni cilj konkurentnog programiranja, a ostali su: **bolje iskorišćenje računara i njegovo čvršće sprezanje sa okolinom.** Dodatni ciljevi ne smeju ugroziti osnovni cilj.

35. Zašto operacija *condition_variable::wait( )* predstavlja prikriveni kraj isključivog regiona?

Jer ta operacija vraća propusnicu i izaziva preključivanje na novu nit koja može da dobije propusnicu i uđe u isključivi region iste deljene promenljive. Programer treba da proceni kada je momenat za pozivanje ove funkcije odgovarajuć.

36. Šta je mrtva petlja?

Mrtva petlja je **pojava međuzavisnosti niti** koja ugrožava upotrebljivost konkurentnih programa. Ona **dovodi do trajnog zaustavljanja aktivnosti niti** što dovodi to toga da izvršavanje programa nema kraj.

Lekcija 3

1. **Šta karakteriše semafor?**

Sinhronizacija niti koju omogućuje semafor se zasniva na zaustavljanju aktivnosti niti, kao i na omogućavanju nastavljanja njihove aktivnosti. **Ulazak niti u kritičnu sekciju zavisi od stanja semafora**. Pri ulasku niti u kritičnu sekciju, semafor se prevodi u stanje koje onemogućuje ulazak druge niti u kritičnu sekciju. Pri izlasku niti iz kritične sekcije, semafor se prevodi u stanje koje dozvoljava novi ulazak u kritičnu sekciju. SEMAFORI SE OBIČNO IMPLEMENTIRAJU U OKVIRU OPERATIVNOG SISTEMA I NJIHOVA IMPLEMENTACIJA SE OBIČNO ZASNIVA NA ONEMOGUĆENJU PREKIDA.

2. **Koje operacije su vezane za semafor?**

Operacije *stop( ) i resume( ).* Operacija ***stop( )* poziva se na početku kritične sekcije i proverava da li je moguć ulaz u kritičnu sekciju**. Ako je moguć ulazak u okviru nje se menja stanje semafora radi onemogućenja novog ulaska u kritičnu sekciju. Ako ulazak nije moguć aktivnost niti se zaustavlja. Operacija ***resume( )* poziva se na kraju kritične sekcije radi promene stanja semafora i omogućavanja da jedna od niti koje čekaju uđe u kritičnu sekciju**.

3. **Kako semafor obezbeđuje sinhronizaciju međusobne isključivosti?**

Stanje semafora sadržano je u polju *state*. Kada je ono pozitivno moguć je ulazak u kritičnu sekciju (početna vrednost je 1). Postoji **polje *queue*** koje **omogućuje zaustavljanje aktivnosti niti** (i njen kasniji nastavak) **u slučaju kada je *state* negativno.**

4. **Kako se obično implementira semafor?**

Obično se implementira u okviru operativnog sistema i tada se njihova implementacija zasniva na kratkotrajnom onemogućenju prekida.

5. **U čemu se semafori razlikuju od isključivih regiona?**

Predstavljaju **dva različita pristupa sinhronizaciji. Isključivi regioni su prilagođeni objektno orjentisanom programiranju, a semafori procedurnom.**

6. **Koji semafori postoje?**

Binarni (čije stanje ne može preći vrednost 1), posebna vrsta binarnog – **raspodeljeni binarni** i **generalni semafor**.

7. Šta karakteriše binarni semafor?

Njegovo **stanje ne može preći vrednost 1**, **omogućuje ostvarenje sinhronizacije međusobne isključivosti ako se stanje inicijalizuje na 1**, operacije ***stop( ) i resume( )*** **su slične operacijama *lock( ) i unlock( )* klase *mutex***. Ako se stanje inicijalizuje na 0, operacije *stop( ) i resume( )* su slične operacijama *wait( ) i notify_one( )* klase *condition_variable*. Pri korišćenju semafora sa inicijalizovanim stanjem 0, mora se obezbediti isključivost kritičnih sekcija drugim semaforom sa inicijalizovanim stanjem 1 što dovodi do mrtve petlje.

8. Šta karakteriše raspodeljeni binarni semafor?

Zbog problema iz prethodnog pitanja, uvodi se **posebna vrsta binarnog semafora** nazvana raspodeljeni semafor. **Realizuje se pomoću više binarnih semafora čija suma stanja ne može preći vrednost 1**. **Pomoću njega se realizuje uslovna sinhronizacija** tako što **se na ulazu u kritičnu sekciju poziva operacija *stop( ),* a na izlazu iz nje *resume( )* tog ili nekog drugog semafora**, tako se najviše jedna nit može nalaziti u jednoj kritičnoj sekciji.

9. Šta karakteriše generalni semafor?

Stanje semafora može sadržati vrednost veću od 1. Omogućuje ostvarenje uslovne sinhronizacije prilikom rukovanja resursima. Pozitivno stanje može predstavljati broj slobodnih primeraka nekog resursa, **zauzimanje primeraka resursa** se opisuje operacijom ***stop( ),* a oslobađanje** operacijom ***resume( ).***

10. Šta omogućuje raspodeljeni binarni semafor?

Omogućuje uslovnu sinhronizaciju.

11. Šta omogućuje binarni semafor?

Omogućuje međusobnu isključivost inicijalizacijom stanja na 1.

12. Šta omogućuje generalni semafor?

Omogućuje uslovnu sinhronizaciju jer njegovo stanje pokazuje broj slobodnih resursa (inicijalizuje se na 0)**.**

13. Koje su prednosti i mane semafora?

Konkurentno programiranje se često oslanja samo na mehanizam semafora za šta je dovoljno samo ponuditi biblioteku sa definicijom semafora i operacijama za rukovanje semaforima. **Njihov mehanizam je jednostavan i efikasan, a mana je što raspodeljeni binarni semafori nisu najpodesnije sredstvo za opisivanje uslovne sinhronizacije.**

lekcija 8

1. **Na šta ukazuje ime datoteke?**

Ime datoteke ukazuje na:

- **njen konkretan sadržaj**
- **na vrstu njenog sadržaja (tip)** - radi klasifikacije datoteka po njihovom sadržaju.

2. **Od koliko delova se sastoji ime datoteke?**

Ime datoteke se sastoji iz **DVA** dela. Prvi deo ukazuje na njen sadržaj, a drugi na vrstu njenog sadržaja, odnosno njen tip (npr. *godina1.txt*).

3. **Od koliko delova se sastoji ime imenika?**

Ime imenika se sastoji iz **JEDNOG** dela (npr. *odsek*).

4. **Šta obuhvata rukovanje datotekom?**

Rukovanje datotekom obuhvata:

- **rukovanje njenim sadržajem**
- **rukovanje njenim imenom**.

5. **Šta karakteriše hijerarhijsku organizaciju datoteka?**

Na VIŠEM nivou hijerarhije su IMENICI, a na NIŽEM nivou se nalaze DATOTEKE. Datoteke pripadaju imenicima, odnosno njihova imena su sadržana u ovim imenicima.

6. **Šta važi za apsolutnu putanju?**

APSOLUTNA PUTANJA, sa svim prethodećim imenicima, **se navodi kad god je moguć nesporazum**, zbog datoteka sa istim imenima, odnosno zbog imenika sa istim imenima.

7. **Šta važi za relativnu putanju?**

Kada postoji mogućnost određivanja RADNOG IMENIKA (njegova putanja se podrazumeva, ne mora se navoditi) koristi se RELATIVNA PUTANJA.

8. **Koje datoteke obrazuju sistem datoteka?**

Sistem datoteka obrazuju **datoteke koje pripadaju istoj hijerarhijskoj organizaciji**.

9. **Koja su prava pristupa datotekama?**

Prava pristupa datotekama su:

- pravo **ČITANJA**,
- pravo **PISANJA**,
- pravo **IZVRŠAVANJA**.

10. Koje kolone ima matrica zaštite?

Matrica zaštite sadrži kolone:

- **vlasnik**
- **saradnik**
- **ostali**.

11. Čemu je jednak broj redova matrice zaštite?

Broj REDOVA matrice zaštite jednak je **broju DATOTEKA.**

12. Gde se mogu čuvati prava pristupa iz matrice zaštite?

Prava pristupa matrice zaštite se mogu čuvati :

- **u DESKRIPTORIMA DATOTEKA** (ako su vezana za datoteke)
- **mogu ih čuvati POJEDINI KORISNICI** (ako su vezana za korisnike).

13. Šta je potrebno za sprečavanje neovlašćenog menjanja matrice zaštite?

Za sprečavanje neovlašćenog menjanja matrice zaštite potrebno je:

- **za svaku datoteku znati ko je njen vlasnik**
- **razlikovanje korisnika**

  To se postiže tako što svaku aktivnost svaki korisnik započinje svojim PREDSTAVLJANJEM (***login***).

14. Kada korisnici mogu posredno pristupiti spisku lozinki?

Korisnici mogu posredno pristupiti spisku lozinki:

- **radi njihovog predstavljanja**
- **radi izmene njihove lozinke.**

15. Koju dužnost imaju administratori?

Dužnosti administratora su **rukovanja spiskovima imena i lozinki registrovanih korisnika** (ubacivanje/izbacivanje parova imena i lozinki iz ovih spiskova).

16. Šta sadrži numerička oznaka korisnika?

Numerička oznaka korisnika sadrži **DVA REDNA BROJA**:

- **redni broj korisnika** (*UID, User IDentification*)
- **redni broj grupe** (*GID, Group Identification*).

17. Kakvu numeričku oznaku imaju saradnici vlasnika datoteke?

SARADNICI vlasnika datoteke imaju **ISTI REDNI BROJ GRUPE kao vlasnik**.

18. Kakvu numeričku oznaku imaju ostali korisnici?

OSTALI korisnici **imaju REDNI BROJ GRUPE koji je RAZLIČIT od rednog broja grupe vlasnika**.

19. Kada se obavlja provera prava pristupa datoteci?

Provera prava pristupa se obavlja **PRE PRVOG PRISTUPA datoteci**.

20. Čime se bavi sigurnost?

Sigurnost se odnosi na uspešnost zaštite od neovlašćenog korišćenja datoteka i ostalih delova računara kojima upravlja operativni sistem. Sigurnost se bavi:

- **Načinima prepoznavanja ili identifikacije korisnika** (*authentiication*)
- **Načinima provere njihovih prava pristupa** (*authorization*)

lekcija 9

1. Šta omogućuju sistemske operacije za rukovanje procesima?

Sistemske operacije za rukovanje procesima obezbeđuju **stvaranje** i **uništavanje procesa**.

(Navedene su osnovne operacije, a postoje još i sistemske operacije za izmenu atributa procesa.)

2. Šta obuhvata stvaranje procesa?

Stvaranje procesa obuhvata **stvaranje njegove slike i njegovog deskriptora, kao i pokretanje njegove aktivnosti.**

3. Šta obuhvata uništenje procesa?

Uništenje procesa obuhvata **zaustavljanje njegove aktivnosti, kao i uništenje njegove slike i njegovog deskriptora.**

4. Šta sadrži slika procesa?

Slika procesa obuhvata **niz lokacija radne memorije sa uzastopnim (logičkim) adresama**. Ona sadrži **izvršavane mašinske naredbe, promenljive i stek**.

5. Za šta se koristi slobodna radna memorija procesa?

Ona je na raspolaganju procesu **za širenje (punjenje) steka, ali i za stvaranje dinamičkih promenljivih** (deo slobodne radne memorije procesa, koji se koristi za stvaranje dinamičkih promenljivih, se na engleskom zove **HEAP**).

6. Koji atributi procesa postoje?

Atributi karatkerišu aktivnost procesa i oni obuhvataju:

1) **stanje procesa** ('spreman', 'aktivan', 'čeka')
2) **sadržaje procesorskih registara** (zatečene u njima pre poslednjeg preključivanja procesora sa procesa)
3) **numeričku oznaku vlasnika procesa**
4) **oznaku procesa stvaraoca**
5) **trenutak pokretanja aktivnosti procesa**
6) **ukupno trajanje aktivnosti procesa** (odnosno, ukupno vreme angažovanja procesora)
7) **podatke o slici procesa** (njenoj veličini i njenom položaju u radnoj i masovnoj memoriji)
8) **podatke o datotekama koje proces koristi**
9) **podatak o radnom imeniku procesa**
10) **razne podatke** neophodne za upravljanje aktivnošću procesa (na primer, prioritet procesa ili položaj sistemskog steka procesa)

7. Koje sistemske operacije za rukovanje procesima postoje?

Postoje 3 grupe sistemskih operacija:

1) sistemske operacije **za stvaranje** procesa (*fork( )* i *exec( )*)
2) sistemske operacije **za uništenje** procesa (*exit( )*)
3) sistemske operacije **za izmenu atributa** procesa

8. Koji se atributi nasleđuju od procesa stvaraoca prilikom stvaranja procesa?

Jedan deo atributa stvaranog procesa se ne mora navoditi prilikom poziva sistemske operacije poziva, jer se taj deo atributa nasleđuje iz deskriptora procesa stvaraoca, kao na primer:

1) numerička oznaka vlasnika procesa
2) podatak o radnom imeniku procesa
3) prioritet procesa

9. Koji atributi procesa nastanu prilikom njegovog stravaranja?

Deo atributa procesa nastanje prilikom njegovog stvaranja, kao na primer **podaci o slici procesa**.

10. U kojim stanjima može biti proces stvaraoc nakon stvaranja novog procesa?

Proces stvaraoc može biti u nekom od sledećih stanja nakon stvaranja novog procesa:

1) '**spreman**' – kada u okviru stvaranja procesa dolazi do preključivanja procesora sa procesa stvaraoca na stvarani proces, tj. kada je prioritet stvaranog procesa viši od prioriteta procesa stvaraoca

2) '**čeka**' – kada proces stvaraoc pozove posebnu sistemsku funkciju *wait( )* da zatraži zaustavljanje svoje aktivnosti i tako omogući preključivanje procesora na stvarani proces (čeka se kraj aktivnosti stvorenog procesa)

3) '**aktivan**' – (ovo nije navedeno u knjizi, ali možda je potrebno naglasiti da se proces stvaralac nalazi u stanju aktivan kada je pri stvaranju procesa njegov prioritet viši od prioriteta stvorenog procesa; takođe, u ovom stanju se nalazi i kada se nastavi njegova aktivnost posle uništenja stvorenog procesa)

11. Šta je stepen multiprogramiranja?

Stepen multiprogramiranja je **najveći mogući broj slika procesa koje mogu istovremeno postojati u radnoj memoriji**.

Što je stepen multiprogramiranja viši, to je i veća verovatnoća da je procesor zaposlen, jer je veća verovatnoća da postoji spreman proces. Jasno je da stepen multiprogramiranja zavisi od veličine radne memorije.

12. Šta karakteriše kopiju procesa?

Kopiju procesa karakteriše **njegova slika u radnoj memoriji**, kao **i kopija slike** koja se nalazi **u masovnoj memoriji**. Slika procesa i njena kopija nastaju istovremeno ili pri prvom izbacivanju slike procesa, a nestaju istovremeno.

(nisam siguran za ovo pitanje, bilo bi dobro ga pročitate i proverite u knjizi)

13. Koje raspoređivanje je vezano za zamenu slika procesa?

U vezi sa zamenom slika procesa jeste **dugoročno raspoređivanje**, u okviru koga se odabira proces čija slika se izbacuje, kao i proces čija slika se ubacuje.

Važno je uočiti da se dugoročno raspoređivanje razlikuje od običnog ili kratkoročnog raspoređivanja, koje među spremnim procesima odabira proces na koji se prekljućuje procesor.

14. Šta karakteriše rukovanje nitima unutar operativnog sistema?

Kada se nitima rukuje unutar operativnog sistema, operativni sistem nudi sistemske operacije za rukovanje nitima, koje omogućuju **stvaranje, uništavanje i sinhronizaciju niti**. U ovom slučaju, deskriptori i sistemski stek niti se nalaze **u sistemskom prostoru**, dok se sopstveni stek niti nalazi u korisničkom prostoru (unutar slike procesa).

15. Šta karakteriše rukovanje nitima van operativnog sistema?

U slučaju kada rukovanje nitima nije u nadležnosti operativnog sistema, **brigu o nitima potpuno preuzima konkurentna biblioteka**. Pošto ona pripada slici procesa, **rukovanje nitima** se u ovom slučaju u potpunosti odvija **u korisničkom prostoru**, u kojem se nalaze i deskriptori niti i stekovi niti.

Osnovna prednost rukovanja nitima van operativnog sistema jeste efikasnost, jer su pozivi potprograma konkurentne biblioteke brži (kraći) od poziva sistemskih operacija.

lekcija 10

1. Šta karakteriše nulti proces ?

Procesor se prekljućuje na nulti proces kada ne postoji drugi spreman proces.

Zadatak beskonačnog procesa je **da zaposli procesor**, **kada nema mogućnosti za korisnu upotrebu procesora**.

U toku aktivnosti beskonačnog procesa izvršava se beskonačna petlja što znači da je **proces uvek spreman ili aktivan.**

Prioritet beskonačnog procesa je **niži od svih ostalih procesa**.

Postoji za **sve vreme** aktivnosti operativnog sistema.

2. Šta je karakteristično za proces dugoročni raspoređivač ?

Dugoročni raspoređivač se **brine o zameni slika procesa** kada za to ima potrebe.

On se periodično aktivira radi povezivanja operacije za zamenu slika procesa. Nakon toga proces se uspava do sledeceg aktiviranja.

Za uspavljivanje se poziva odgovarajuća sistemska operacija koja pripada sloju za rukovanje kontrolerima.

Postoji za **sve vreme** aktivnosti operativnog sistema.

3. Šta radi proces identifikator ?

Proces identifikator **podržava predstavljanje korisnika.**

Proces identifikator **opslužuje terminal**, da bi posredstvom njega stupio u interakciju sa korisnikom u toku predstavljanja, radi **preuzimanja imena**(prikazuje se na ekranu) **i lozinke**(ne prikazuje se na ekranu) korisnika.

4. Ko stvara proces komunikator ?

Po preuzimanju imena i lozinke , **proces identifikator** proverava njihovu ispravnost, i ako je prepoznao korisnika tada stvara proces komunikator**.**

5. Šta sadrži slog datoteke lozinki ?

Slog datoteke lozinki sadrži :

a) **Ime i lozinku korisnika**

b) **Numeričku oznaku korisnika**

c) **Putanju radnog imenika korisnika**

d) **Putanju izvršne datoteke**, sa inicijalnom slikom korisničkog procesa komunikatora

6. Šta označava SUID (switch user identification) ?

Ako je administrator vlasnik izvršne datoteke sa inicijalnom slikom procesa za izmenu lozinki, dovoljno je naznačiti da on treba da bude vlasnik i procesa nastalog na osnovu ove izvršne datoteke.

7. Šta je neophodno za podmetanje trojanskog konja ?

Korisnici moraju biti oprezni da sami ne odaju svoju lozinku procesu identifikatoru. To se može desiti, ako se njihov prethodnik ne odjavi, nego ostavi svoj proces da opslužuje terminal, oponašajući proces identifikator. Ovakvi procesi se nazivaju trojanski konji.

8. Šta karakteriše simetričnu kriptografiju ?

Ako **algoritam dekriptovanja direktno i jednoznačno sledi iz algoritma kriptovanja, a ključ dekriptovanja iz ključa kriptovanja** tada je reč o simetričnoj kriptografiji.
Osobina simetrične kriptografije je da **poznavanje ključa kriptovanja omogućuje i dekriptovanje**. Zato kod simetrične kriptografije ključ kriptovanja predstavlja tajnu kao i ključ dekriptovanja.
Simetrična kriptografija nije podesna za kriptovanje poruka, jer tada ključ kriptovanja mora znati svaki pošiljalac poruke, što ga dovodi u poziciju da moze da dekriptuje poruke drugih pošiljalaca.

9. Šta karakteriše asimetričnu kriptografiju ?

Osobina asimetrične kriptografije je da se iz ključa kriptovanja ne može odrediti ključ dekriptovanja, pa **poznavanje kriptovanja ne omogućuje dekriptovanje**.
Asimetrična kriptografija se temelji na korišćenju jednostavnih algoritama kriptovanja kojima odgovaraju komplikovani algoritmi dekriptovanja, zato je asimetrična kriptografija **mnogo sporija od simetrične.**

10. Na čemu se temelji tajnost kriptovanja ?

Tajnost kriptovanja teksta se zasniva na činjenici da (1) **komplikovanost algoritama kriptovanja i dekriptovanja**, (2) **velika dužina ključeva kriptovanja i dekriptovanja**, kao i (3) **veliki broj ovih ključeva** čine praktično neizvodljivim pokušaj da se dekriptuje kriptovani tekst probanjem, jednog po jednog, svih mogućih ključeva dekriptovanja.

lekcija 11

1. Kako se predstavlja sadržaj datoteke?

Sadržaj datoteke se predstavlja **kao niz bajta** koji se može produžavati ili skraćivati po potrebi. Bajtima iz niza se može pristupati u proizvoljnom redosledu, korišćenjem rednog broja bajta za njegovu identifikaciju.

2. Gde se javlja interna fragmentacija?

INTERNA FRAGMENTACIJA se javlja **kod POSLEDNJEG BLOKA datoteke** koji predstavlja neupotrebljen deo masovne memorije.

3. Šta karakteriše kontinualne datoteke?

KONTINUALNE datoteke karakterise **operacija PREBACIVANJA**. Da bi prebacivanje bilo moguće neophodno je da sloj za rukovanje datotekom uspostavi preslikavanje bajta u redne brojeve njima odgovarajućih blokova. Najjednostavnije je ako se sadržaj datoteke nalazi u SUSEDNIM blokovima.

4. Koji oblik evidencije slobodnih blokova masovne memorije je podesan za kontinualne datoteke?

Za KONTINUALNE datoteke je podesna **EVIDENCIJA U OBLIKU NIZA BITA** u kome svaki bit odgovara jednom bloku i pokazuje da li je on zauzet (0) ili slobodan (1).

5. Šta je eksterna fragmentacija?

EKSTERNA FRAGMENTACIJA je **pojava iscepkanosti slobodnih blokova masovne memorije u kratke nizove susednih blokova** koja otežava rukovanje kontinualnim datotekama. Ona nastaje kao rezultat višestrukog stvaranja i uništavanja datoteka u slučajnom redosledu, pa nakon uništavanja datoteka ostaju nizovi slobodnih susednih blokova, međusobno razdvojeni blokovima postojećih datoteka.

6. Šta karakteriše rasute datoteke?

RASUTE datoteke karakteriše to da je kod njih sadržaj **smešten u NESUSEDNIM blokovima masovne memorije**. Kod rasutih datoteka **REDNI BROJEVI BAJTA se preslikavaju u REDNE BROJEVE BLOKOVA** pomoću TABELE PRISTUPA.

7. Šta karakteriše tabelu pristupa?

Tabela pristupa **služi za preslikavanje rednih brojeva bajta u redne brojeve blokova kod rasutih datoteka**. **Elementi** TABELE PRISTUPA **sadrže REDNE BROJEVE BLOKOVA**. Indekse ovih elemenata određuje količnik rednog broja bajta i veličine tabele pristupa. Veličina tabele pristupa ograničava dužinu rasutih datoteka.

8. Šta ulazi u sastav tabele pristupa?

U sastav tabele pristupa ulaze:

- **POČETNI ODSEČAK** (uvek je prisutan, ima p elemenata tabele pristupa),
- **DODATNI ODSEČCI** (prisutni su samo kada su neophodni, imaju b elemenata tabele pristupa, b>p),
- **BLOK PRVOG STEPENA** (ima do b rednih brojeva blokova sa dodatnim odsečcima, u svakom od njih se nalazi b novih elemenata tabele pristupa),
- **BLOK DRUGOG STEPENA** (ima do b rednih brojeva blokova prvog stepena, svaki od njih sadrži do b rednih brojeva blokova sa dodatnim odsečcima, a u svakom od njih se nalazi b novih elemenata tabele pristupa).

9. Kada rasuta datoteka ne zauzima više prostora na disku od kontinualne datoteke?

10. Koji oblik evidencije slobodnih blokova masovne memorije je podesan za rasute datoteke?

Evidencija slobodnih blokova masovne memorije je **u obliku LISTE SLOBODNIH BLOKOVA**.

11. Kada dolazi do gubitka blokova prilikom produženja rasute datoteke?

Do gubitka blokova prilikom produženja rasute datoteke dolazi **prilikom**:

- **izmene evidencije slobodnih blokova**, radi isključenja pronađenog slobodnog bloka iz ove evidencije,
- **izmene tabele pristupa produžavane rasute datoteke**, radi smeštanja rednog broja novog bloka u element ove tabele.

12. Kada dolazi do višestrukog nezavisnog korišćenja istog bloka prilikom produženja rasute datoteke?

Prilikom IZMENE TABELE PRISTUPA PRODUŽAVANE RASUTE DATOTEKE, radi smeštanja rednog broja novog bloka u element ove tabele dolazi do istovremenog uključivanja istog bloka u više rasutih datoteka.

13. Kada pregled izmena ukazuje da je sistem datoteka u konzistentnom stanju?

Ako u pregledu izmena nije registrovan potpun opis nameravane izmene, tada izmena nije ni započeta, pa je sistem datoteka u konzistentnom stanju.

14. Kako se ubrzava pristup datoteci?

Pristup datoteci se ubrzava **zauzimanjem prostora u radnoj memoriji za više bafera**, namenjenih za čuvanje kopija korišćenih blokova.

15. Od čega zavisi veličina bloka?

Na veličinu bloka utiču:

- **brzina prebacivanja podataka između radne i masovne memorije,**
- **interna fragmentacija.**

16. Šta sadrži deskriptor kontinualne datoteke?

Deskriptor kontinualne datoteke, pored atributa koji omogućuju preslikavanje rednih brojeva bajta u redne brojeve blokova, sadrži i:

- **numeričku oznaku** vlasnika datoteke,
- **prava pristupa datoteci** za njenog vlasnika, za njegove saradnike i za ostale korisnike,
- **podatak da li je datoteka zaključana ili ne**,
- **SUID podatak** da li numerička oznaka vlasnika datoteke postaje numerička oznaka vlasnika procesa stvorenog na osnovu sadržaja datoteke (važi samo za izvršene datoteke),
- **datum poslednje izmene datoteke.**

17. Kako se rešava problem eksterne fragmentacije?

Problem eksterne fragmentacije se može rešiti **SABIJANJEM DATOTEKA**, tako da svi slobodni blokovi budu potisnuti iza datoteka i da tako obrazuju niz susednih blokova.

18. Kako se ublažava problem produženja kontinualne datoteke?

Problem produženja datoteke se ublažava, **ako se dozvoli da se kontinualna datoteka sastoji od više kontinualnih delova.** Pri tome se za svaki od ovih delova u deskriptoru datoteke čuvaju podaci o rednom broju početnog bloka dotičnog dela i o njegovoj dužini.

19. Šta sadrži deskriptor rasute datoteke?

Deskriptor rasute datoteke sadrži:

- **početni odsečak tabele pristupa,**
- **redni broj njenog prvog dodatnog odsečka,**
- **redni broj bloka prvog stepena indirekcije,**
- **redni broj bloka drugog stepena indirekcije,**
- **dužinu rasute datoteke.**

20. Šta je imenik?

Imenik je **datoteka** koja **sadrži tabelu** u čijim elementima su **imena datoteka** (*imena imenika*) i **redni brojevi njihovih deskriptora**.

21. Šta sadrže elementi tabela otvorenih datoteka?

Elementi tabela otvorenih datoteka sadrže **ADRESU KOPIJE DESKRIPTORA** odgovarajuće datoteke iz tabele deskriptora datoteka.

22. Koje sistemske operacije za rukovanje datotekama postoje?

Sistemske operacije za rukovanje datotekama su:

- operacija za **OTVARANJE** datoteke – *open()*,
- operacija za **STVARANJE** datoteke - *create()*,
- operacija za **ZATVARANJE** datoteke - *close()*,
- operacija **KOJA RUKUJE KOPIJOM DESKRIPTORA ZAKLJUČAVANE/OTKLJUČAVANE** datoteke - *flock()*,
- operacije **ČITANJA** datoteke – *read()*,
- operacija **PISANJA** datoteke – *write()*,
- operacija **IZMENE POZICIJE** datoteke – *seek()*,
- operacija za **IZMENU ATRIBUTA** datoteke,
- operacija za **STVARANJE LINKA** – *link()*,
- operacija za **UNIŠTENJE** datoteke – *unlink()*,
- operacija za **SPAJANJE DVA SISTEMA DATOTEKA** – *mount()*,
- operacija za **RAZDVAJANJE DVA SISTEMA DATOTEKA** –*umount()*.

23. Koji su argumenti sistemskih operacija za rukovanje datotekama?

Argumenti sistemskih operacija su:

- **za operaciju OTVARANJA** - **PUTANJA** datoteke, **OZNAKA NAMERAVANE VRSTE PRISTUPA** otvaranoj datoteci
- **za operaciju ZATVARANJA** - **INDEKS OTVORENE DATOTEKE**,
- **za operaciju ZAKLJUČAVANJA** - **INDEKS OTVORENE DATOTEKE**,
- **za operacije ČITANJA I PISANJA** - **INDEKS OTVORENE datoteke i INDEKS BAJTA**,
- samo **za operaciju ČITANJA**- **adresa zone radne memorije u U KOJU SE SMEŠTAJU PROČITANI BAJTI,**
- samo **za operaciju PISANJA – adresa zone radne memorije IZ KOJE SE PREUZIMAJU BAJTI ZA PISANJE,**
- **za operaciju IZMENE POZICIJE DATOTEKE** - **INDEKS OTVORENE datoteke i PODATAK O NOVOJ POZICIJI,**
- **za operacije ZA IZMENU ATRIBUTA DATOTEKE** - **PUTANJA** datoteke, **NOVA VREDNOST** menjanog atributa datoteke,
- **za operaciju STVARANJA LINKA – PUTANJA SA IMENOM** datoteke, **PUTANJA SA LINKOM,**
- **za operaciju UNIŠTENJA**- **PUTANJA UNIŠTAVANE datoteke.**

24. Šta karakteriše specijalne datoteke?

Specijalne datotke karakteriše to da **predstavljaju pojedine ULAZNE ili IZLAZNE uređaje**. One se dele na **ZNAKOVNE** (tastatura, ekran, štampač, mrežni kontroler) **i BLOKOVSKE** (diskovi).

25. Šta sadrži deskriptor specijalne datoteke?

Deskriptori specijalnih datoteka **sadrže ATRIBUTE**, kao što su:

- **numerička oznaka vlasnika datoteke,**
- **prava pristupa datoteci** za njenog vlasnika, za njegove saradnike i za ostale korisnike,
- **podatak da li je datoteka zaključana** ili ne.

26. Šta omogućuju blokovske specijalne datoteke?

Blokovske specijalne datoteke **omogućuju direktne pristupe blokovima diska** (važno je kod pronalaženja izgubljenih blokova, kod sabijanja datoteka, ili kod pripremanja disk jedinica za korišćenje).

27. Šta omogućuje rukovanje particijama?

Rukovanje particijama **dozvoljava formiranje logičkih disk jedinica** koje obuhvataju više particija na raznim fizičkim jedinicama. **To omogućava**:

- **BRŽI PRISTUP BLOKOVIMA** logičke disk jedinice (istovremeno mogu biti prebacivani blokovi iz raznih particija sa raznih fizičkih diskova) ili
- **VEĆU POUZDANOST** logičke disk jedinice (svaki blok može biti repliciran tako da svaka partiicja sa raznih fizičkih diskova sadrži potpunu kopiju logičke disk jedinice) ili
- **OBOJE**.

lekcija 12

1. Kakav može biti logički adresni prostor?

Logički adresni prostor može biti: **kontinualan**, sastavljen od **segmenata različite veličine**, sastavljen od **stranica iste veličine** i sastavljen od **segmenata raznih veličina** koji se sastoje od **stranica iste veličine**.

2. Šta karakteriše kontinualni logički adresni prostor?

Kontinualni logički prostor se sastoji od **jednog niza uzastopnih logičkih adresa**, koje počinju od 0. **Veličina** kontinualnog logičkog adresnog prostora **nadmašuje potrebe** prosečnog procesa. Da bi se detektovao pokusaj **izlaska procesa** iz njegovog adresnog prostora, koristi **se granična adresa**. **Dodavanjem bazne adrese logičkoj adresi nastaje fizička adresa**, gde bazna adresa predstavlja adresu od koje počinje niz fizickih adresa.

3. Šta karakteriše segmentirani logički adresni prostor?

Segmentirani logički adresni prostor **se sastoji od više nizova uzastopnih logičkih adresa.** Da bi se znalo kom segmentu pripada logička adresa, njeni **značajniji** biti sadrže **adresu njenog segmenta**, a **manje** značajni **unutrašnju adresu** u segmentu. Svaki **segment** ima svojstva **kontinualnog** adresnog logičkog prostora, pa ga karakterišu njegova bazna i granična adresa. **Za translaciju** je potrebna **tabela segmenata**, gde svaki element sadrži grancinu i baznu adresu.

4. Šta karakteriše stranični logički prostor?

Stranični logički prostor se sastoji od **jednog niza uzastopnih logičkih adresa, podeljenog u stranice iste veličine**. Da bi se znalo kojoj stranici pripada adresa, **značajniji** biti sadrže **adresu stranice**, a **manje** značajni **unutrašnju adresu** u stranici. Svaka **stranica je kontinualna** pa ima svojstva kontinualnog adresnog logičkog prostora. Jedina razlika je da je **veličina stranice unapred poznata** i jednaka stepenu broja 2, zbog čega je dovoljna samo bazna adresa. **Za translaciju** je potrebna **tabela stranica**, gde svaki element sadrži baznu adresu odgovarajuće stranice.

5. Šta karakteriše stranično segmentirani logički adresni prostor?

Stranično segmentiran adresni prostor se sastoji od **više nizova uzastopnih logičkih adresa**, za svaki segment po jedan **niz, podeljen u stranice iste veličine. Najznačajniji** biti sadrže **adresu segmenta**, **srednji** biti sadrže **adresu stranice**, **i najmanje** značajni sadrže **unutrašnju** adresu. Segment je kontinualan, kao i stranica s tim da je njena veličina unapred poznata i jednaka stepenu broja 2. **Za translaciju je potrebna tabela segmenata i više tabela stranica**. Elementi tabele segmenata sadrže granične adrese stranice u segmentu i baznu adresu tabele stranica segmenata. Adresa segmenata indeksira element tabele segmenata.

6. Šta karakteriše translacione podatke?

Translacioni podaci obuhvataju ili **graničnu i baznu adresu** ili **tabelu segmenata** ili **tabelu stranica** ili **tabelu segmenata sa pripadnim tabelama stranica**. Njih operativni sistem priprema prilikom stvaranja procesa. Oni se menjaju prilikom preključivanja.

7. Šta karakteriše translaciju logičkih adresa kontinualnog logičkog adresnog prostora u fizičke adrese?

Da bi translacija bila moguća, neophodno je da se u lokacije **fizičke radne memorije** smesti **slika procesa**. Tada translacija odgovara dinamičkoj relokaciji i olakšava zamenu slike procesa. Pokušaj izlaska izaziva izuzetak na koji reaguje OS.

8. Koji logički adresni prostor se koristi kada veličina fizičke radne memorije prevazilazi potrebe svakog procesa?

Koristi **se kontinualni adresni logički prostor**.

9. Šta karakteriše segmentaciju?

Segmentacija se koristi kada je **važno racionalno korišćenje fizičke radne memorije**. **Segmenti** logičkog adresnog prostora **sadrže delove** njegove **slike**. Ako istovremeno postoji više procesa nastalih od istog programa, oni mogu **deliti iste mašinske naredbe**. Omogućeno je **preslikavanje unutrašnjih adresa** segmenta naredbi raznih procesa na **iste fizičke adrese**. Postoji i mogućnost **naknadnog proširenja** segmenta. **Segmentacija ubrzava zamenu slika procesa**. **Segmentacija** je **puna** ako se dozvoljava da **svakom podprogramu odgovara 1 segment**. Tada se vodi računa o sinhronizaciji radi ocuvanja konzistentnosti deljenih promenljivih.

10. Koji logički adresni prostor se koristi kada je važno racionalno korišćenje fizičke radne memorije?

Segmentacija se koristi kada je vazno racionalno koriscenje fizicke radne memorije.

11. Koji logički adresni prostor se koristi kada je veličina fizičke radne memorije nedovoljna za pokrivanje potreba tipičnog procesa?

Stranični logički adresni prostor se koristi kada je logički adresni prostor procesa veći od raspoloživog fizičkog. On se naziva i virtuelni.

12. Šta sadrže elementi tabele stranica?

Broj elemenata određuje najveci broj stranica u straničnom logičkom adresnom prostoru. Svaki element sadrži **baznu adresu odgovarajuće stranice**. Adresa stranice indeksira element tabele stranica, a bazna adresa se koristi za translaciju.

13. Šta karakteriše virtuelni adresni prostor?

Stranični logicki adresni prostor ili virtuelni adresni prostor pripada virtuelnoj memoriji i **sadrži virtuelne adrese. Stranice** virtuelnog adresnog prostora se nalaze na **masovnoj memoriji**. Postoji posebna evidencija gde se nalaze jer se za izvršavanje mašinskih naredbi u fizičkoj radnoj memoriji moraju nalaziti bajti mašinskog formata i bajti operanda. U **fizičkoj memoriji** se moraju nalaziti samo **kopije** virtuelnih stranica. Praktična upotreba virtuelne memorije se temelji na lokalnosti izvrešenja programa, tj. u fizičkoj memoriji je dovoljno da bude samo deo programa koji se izvršava.

14. Po kom principu se prebacuju kopije virtuelnih stranica?

Kopije se prebacuju **automatski iz masovne u fizičku memoriju i obrnuto**, kada zatreba. Iz fizičke u masovnu se prebacuje kada je potrebno osloboditi lokacije sa kopijama u medjuvremenu izmenjenih virtuelnih stranica. Pored oslobađanja, cilj ovog prebacivanja je da se obezbedi da masovna memorija sadrži azurnu sliku procesa. Registrovanje svake izmene se vrši automatski. **Kopije virtuelnih stranica se prebacuju na zahtev** a ne unapred, jer u opštem slučaju ne postoji način da se predvidi redosled korisćenja virtuelnih stranica.

15. Šta karakteriše straničnu segmentaciju?

Stranična segmentacija **se koristi kada su segmenti potrebni radi racionalnog korišćenja fizičke radne memorije, a njihova veličina nadmašuje veličinu raspoložive fizičke memorije**. U tom slučaju svaki segment uvodi sopstveni virtuelni prostor. Zahvaljujući tome, **stranice virtuelnog adresnog prostora segmenta se nalaze u masovnoj memoriji, a u fizičkim stranicama samo neophodne kopije virtuelnih segmenata.** Stranična segmentacija nudi koncept memorijski preslikane datoteke gde se ne zahteva sistemska operacija za čitanje, pisanje jer se pristupa direktno lokacijama.

16. Koji logički adresni prostor se koristi kada je važno racionalno korišćenje fizicke radne memorije, a ona ima nedovoljnu veličinu?

Straničnо segmentirani logički adresni prostor se koristi kada su segmenti potrebni radi racionalnog koriscenja, a njihova velicina nadmašuje veličinu raspoložive fizičke memorije.

17. Kako se deli fizička radna memorija?

Fizička radna memorija **se deli na lokacije koje stalno zauzima operativni sistem i na preostale slobodne lokacije koje su na raspolaganju za stvaranje procesa** i drugih potreba. Operativni sistem zauzima lokacije sa kraja i sa početka fizičkog adresnog prostora, a one između su slobodne.

18. Kako se deli virtuelni adresni prostor?

Donja polovina, sa nižim adresama virtuelnog prostora se stavlja **na raspolaganje svakom procesu** dok se **gornja polovina** rezerviše **za operativni sistem**. Zbog toga se **gornja polovina** naziva **sistematski virtuelni adresni prostor, a donja polovina** se naziva **korisnički virtuelni prostor**.

19. U kom obliku može biti evidencija slobodne fizičke memorije?

Evidencija može biti **u obliku niza bita**, za potrebe kontinualnog ili segmentiranog adresnog prostora. Mana ove evidencije je dugotrajnost operacija zauzimanja i oslobađanja. Zbog toga se češće evidencija slobodne fizičke radne memorije pravi **u obliku liste slobodnih odsečaka radne memorije**.

20. Kod kog adresnog prostora se javlja eksterna fragmentacija?

Kod kontinualnog ili segmentiranog logickog adresnog prostora se javlja problem eksterne fragmenatcije.

21. Kako se nazivaju skupovi fizičkih stranica, koji se dodeljuju procesima?

Skupovi fizičkih stranica koji se dodeljuju procesima se nazivaju **minimalni skupovi.**

22. Kada treba proširiti skup fizičkih stranica procesa?

Uvećanje skupa ima smisla **samo ako to dovodi do smanjivanja učestanosti straničnih prekida**. Znači, **ukoliko je u toku aktivnosti procesa učestanost straničnih prekida iznad neke gornje granice** tada ima smisla uvecanje skupa.

23. Kada treba smanjiti skup fizičkih stranica procesa?

Ako je učestanost straničnih prekida ispod neke granice, tada ima smisla smanjenje skupa fizičkih stranica procesa, jer i sa manjim skupom stranica učestanost straničnih prekida ostaje u prihvatljivom rasponu.

24. Kada ne treba menjati velicinu skupa fizickih stranica procesa?

Ukoliko je učestanost straničnih prekida između donje i gornje granice, tada nema potrebe za izmenom broja fizičkih stranica u skupu stranica aktivnog procesa. U ovom slucaju skup fizickih stranica obrazuje radni skup.

25. Koji pristupi oslobadjanja fizičkih stranica obezbeđuju smanjenje učestanosti straničnih prekida nakon povećanja broja fizičkih stranica procesa?

Pristupi **pronalaženja fizičke stranice koja nije korišćena u nekom periodu (NRU), pronalaženja najmanje korišćene fizičke stranice (LRU), softversko simuliranje pronalaženja najmanje korišćene stranice (NFU/aging)** obezbeđuju smanjenje učestanosti prekida nakon povećanja broja fizičkih stranica.

26. Koji pristupi oslobađanja fizičkih stranica koriste bit referenciranja?

NRU, NFU, LRU, second chance, clock, wsclock

27. Koji pristupi oslobađanja fizičkih stranica koriste bit izmene?

NRU, FIFO (oslobađanje stranica sa najstarijim sadržajem)

28. Na šta se oslanja rukovanje virtuelnom memorijom?

Sloj za rukovanje virtuelnom memorijom se oslanja **na operacije sloja za rukovanje kontrolerima**, da bi se obezbedio prenos kopija na relaciji masovna-radna memorija i da bi se smestili odgovarajući elementi tabele prekida.

lekcija 13

1. **Šta karakteriše ulazne i izlazne uređaje ?**

Ulazni i izlazni uređaji **se dele na blokovske i znakovne**. **Za blokovske uređaje jedinica pristupa je blok,** kod njih je **dozvoljen direktan pristup i ne dozvoljavaju dinamičko podešavanje** pojedinih funkcionalnih karakteristika. **Kod znakovnih uređaja jedinica pristupa je znak, dozvoljen je sekvencijalni pristup kao i dinamičko podešavanje** funkcionalnih karakteristika.

2. **Koja svojstva imaju drajveri?**

Zajedničko svojstvo drajvera je da je svaki namenjen za rukovanje određenom klasom uređaja. Jedan drajver **može da opsluži više primeraka uređaja iste klase**. Drajveri se nalaze **u tesnoj saradnji sa kontrolerima ulaznih i izlaznih uređaja**. Van drajvera su vidljive samo operacije. Tipične operacije drajvera blokovskih uređaja - inicijalizacija, operacija ulaza/izlaza blokova. Tipične operacije drajvera znakovnih uređaja - inicijalizacija, operacije ulaz/izlaz znakova, upravljačke operacije.

3. **Šta karakteriše tabelu drajvera?**

Tabela drajvera **nudi način povezivanja sloja za rukovanje datotekama i sloja za rukovanje kontrolerima**. **Zahvaljujući njoj moguće je u OS dodavati nove drajvere**. **Za adrese svake od operacija predviđeno je posebno polje u tabeli drajvera**. Redni broj drajvera indeksira element tabele, koji sadrži polja sa adresama operacija. Polja namenjena za adrese operacija koje drajver ne podržava popunjena su lažnim operacijama koje nemaju nikakav efekat.

4. **Šta podrazumeva podela drajvera na gornji i donji deo?**

Za razliku od operacija drajvera, koje se pozivaju iz sloja iznad sloja za rukovanje kontrolerima, obrađivače prekida poziva mehanizam prekida, znači hardver ispod OS. Zato **operacije drajvera obrazuje gornji deo drajvera, a obradjivači prekida dodnji deo drajvera.**

5. **Kada se pozivaju operacije drajvera blokovskih uredjaja?**

Aktivnost drajvera započinje inicijalizacijom njihovih kontrolera kada se poziva operacija inicijalizacije. Nakon toga, **aktivnost se svodi na prenos blokova ka i od pomenutih uređaja kada se pozivaju operacija ulaza i izlaza**. Drajversku operaciju ulaza bloka poziva operacija čitanja sloja za rukovanje datotekama. Drajverska operacija izlaza se poziva radi izmene sadržaja bloka ili oslobađanja bafera, a može biti isprovocirana i sistemskom operacijom pisanja sloja za rukovanje datotekama.

6. Šta sadrži lista zahteva?

Svaki zahtev u listi zahteva, pa time i sama lista mora sadržati:

1.**smer zahtevanog prenosa bloka,**

2.**redni broj ovog bloka**,

3.**adresu bafera koji učestvuje u prenosu**

4.**adresu deskriptora procesa** čija aktivnost se zaustavlja do obavljanja zahtevanog prenosa bloka.

7. Šta spada u nadležnost drajvera blokovskih uređaja, ali i kontrolera?

U nadležnost drajvera spada **određivanje načina preslikavanja blokova u sektore**, mada to može obavljati i kontroler. **O optimizaciji kretanja glave diska** takđe mogu da se brinu i drajveri i kontroler.

8. Kada se uzastopni blokovi preslikavaju u prostorno uzastopne sektore?

Ako kontroler automatski prebacuje sve sektore staze iznad koje se kreće glava diska, u svoju lokalnu radnu memoriju, tada nema smetnje da se blokovi preslikaju u prostorno uzastopne sektore.

9. Na koji drajver se odnosi elevator algoritma?

Drajver blokovskog uređaja kada opslužuje magnetni disk.

10. Koju ulogu imaju sistemski procesi posrednici?

Sistemski procesi posrednici **posreduju u korišćenju** nekih uređaja kao što su **štampaci ili mrežni kontroleri**. Svaki od ovih procesa pristupa svom znakovnom uređaju kao specijalnoj datoteci koju zaključava da bi obezbedio isključivost. Uz svaki od sistemskih procesa posrednika postoji i poseban imenik pa kada korisnički proces štampa tekst, prvo se pripremi odgovarajuća datoteka i ona se dodaje u imenik. Sistemski proces je vadi iz imenika i izvršava. Da bi saradnja između procesa bila moguća, neophodna je sinhronizacija procesa.

11. Kada se specijalna datoteka tipično zaključava?

Kod znakovnih uredjaja svaki od sistemskih procesa posrednika pristupa svom znakovnom uređaju kao specijalnoj datoteci koju zaključava da bi obezbedio isključivost.

12. Šta sadrži drajver terminala?

Za drajver terminala je **potreban par bafera za svaki od terminala** koje drajver opslužuje. **Ulazni bafer** služi za smeštanje znakova prispelih sa tastature a drugi, **eho bafer**, služi za smeštanje znakova upućenih ka ekranu.

13. U kom slučaju nisu potrebni eho bafer i obrađivač prekida ekrana?

Za **grafičke terminale** nije potreban eho bafer niti obrađivač prekida ekrana, jer ovakvi terminali poseduju video memoriju čiji se sadržaj periodično prikazuje prilikom osvežavanja ekrana.

14. Šta omogućuje upravljačka operacija drajvera terminala?

Upravljačka operacija omogućuje da se drajveru terminala saopšti **da interpretira znakove** koji dolaze sa tastature ili da ih **ne interpretira**. Takođe, u sklopu toga, drajver mora da omogući **brisanje** poslednjeg prispelog znaka. On **brine i o interpretaciji upravljačkih znakova**. On može interpretirati znakove radi pomeranja kursora, rukovanja ekranom i prozorima.

15. Koje operacije sadrži gornji deo drajvera sata?

Gornji deo drajvera sata je obrađivač prekida koji broji prekide sata, a njihov zbir predstavlja sistemsko vreme. Obuhvata **sistemske operacije za preuzimanje ili izmenu sistemskog vremena, i za uspavljivanje odnosno odlaganje aktivnosti procesa**.

16. Šta ne može da meri drajver sata?

Drajver sata **ne može da meri trajanje aktivnosti procesa**, jer ono nije precizno. Nju izaziva nemogućnost merenja dužine vremenskih intervala koji su kraći od perioda prekida sata.

17. Šta omogućuje obradjivač prekida iz donjeg dela drajveta sata?

Obradjivač prekida donjeg dela sata omogućuje:

1. **odrzavanje sistemskog vremena**,
2. **praćenje isticanja kvantuma** aktivnog procesa,
3. **praćenje ukupnog korišćenja procesorskog vremena** aktivnog procesa,
4. **proveru da li je nastupilo vreme budjenja uspavanog procesa**,
5. **skupljanje statistike o aktivnosti procesa**.

lekcija 14

1. **Šta karakteriše tipične ciljeve raspoređivanja?**

Tipični ciljevi raspoređivanja su:

- **poboljšanje iskorišćenja procesorskog vremena**
- **ravnomerna raspodela procesorskog vremena**
- **što kraći odziv na korisničku akciju ili neki drugi oblik postizanja potrebnog kvaliteta usluge**

Ovakvi ciljevi **nisu saglasni**, pa se ne mogu istovremeno ostvariti!

2. **Šta je cilj raspoređivanja za neinteraktivno korišćenje računara?**

Cilj raspoređivanja za neinteraktivno korišćenje računara je **poboljšanje iskorišćenja procesorskog vremena.** Ovakav cilj se postiže minimiziranjem preključivanja na neophodan broj.

3. **Šta je cilj raspoređivanja za interaktivno korišćenje računara?**

Ciljevi raspoređivanja za interaktivno korišćenje računara su:

- **ravnomerna raspodela procesorskog vremena između istovremeno postojećih procesa**
- **što kraći odziv na korisničku akciju**

4. **Zašto je uvedeno kružno rasporedjivanje?**

Kružno raspoređivanje je uvedeno **zbog ravnomerne raspodele procesorskog vremena i što kraćeg odziva na korisničku akciju**. Kružno raspoređivanje svakom od istovremeno postojećih procesa dodeljuje **jednak vremenski interval**, nazvan **kvantum**.

Po isticanju kvantuma procesor se preključuje na proces koji najduže čeka na svoj kvantum. Kružno raspoređivanje se koristi i kada hitnost svih procesa nije ista, ali se kružno raspoređivanje primenjuje u okviru grupe procesa sa istim prioritetom. Procesor se preključuje na procese sa nižim prioritetom tek kada se završi aktivnost i poslednjeg procesa sa višim prioritetom.

5. **Šta doprinosi ravnomernoj raspodeli procesorskog vremena?**

Dinamička izmena prioriteta procesa doprinosi ravnomernoj raspodeli procesorskog vremena između procesa, ako se uspostavi obrnuta proporcionalnost između prioriteta procesa i obima u kome je on iskoristio poslednji kvantum. **Lutrijsko raspoređivanje** takođe doprinosi ravnomernoj raspodeli procesorskog vremena. Ono se zasniva na dodeli procesima lutrijskih lozova. Tako, ako ukupno ima m lozova, onda proces koji poseduje n od m lozova (n<m), u proseku koristi n/m kvantuma procesorskog vremena.

6. Šta je cilj raspoređivanja za multimedijalne aplikacije?

Cilj raspoređivanja za multimedijalne aplikacije je **garantovanje procesima potrebnog broja kvantuma u pravilnim vremenskim razmacima.**

7. Do čega dovodi skraćenje kvantuma?

Skraćenje kvantuma doprinosi **poboljšanju odziva**, ali i **smanjenju iskorišćenja procesora**, jer povećava broj preključivanja koja troše procesorsko vreme.

8. Šta se postiže uticanjem na nivo prioriteta i na dužinu kvantuma?

Dinamičkom izmenom nivoa prioriteta procesa moguće je **postizanje dobrog odziva za procese koji su u interakciji sa korisnicima**, kao i **postizanje dobrog iskorišćenja procesora za pozadinske procese** koji nisu u čestoj interakciji sa korisnicima. **Interaktivnim procesima** se dodeljuje **najviši prioritet i najkaraći kvantum**, a **pozadinskim** se dodeljuje **najniži prioritet i najduži kvantum**.

lekcija 15

1. Šta je mrtva petlja ?

Mrtva petlja je problematična **pojava trajnog zaustavljanja aktivnosti međusobno zavisnih procesa.** Npr. ona se javlja kada dva procesa žele da u režimu međusobne isključivosti pristupe dvema datotekama.

2. Po čemu se živa petlja razlikuje od mrtve petlje ?

U slučaju blokirajućih sistemskih oparacija zaključavanja, pokušaj prvog procesa da zaključa drugu datoteku **dovodi do trajnog zaustavljanja njegove aktivnosti**. Isto se dešava i sa drugim procesom. **(MRTVA PETLJA)**

U slučaju neblokirajućih sistema operacije zaključavanja, procesi upadaju u **beskonačnu petlju**, pokušavajući da zaključaju datoteku koju je zaključao drugi proces. **(ŽIVA PETLJA)**

*Ove dve petlje se po svom ishodu suštinski ne razlikuju.

3. Koji uslovi su potrebni za pojavu mrtve potlje ?

Potrebno je da budu ispunjena **4 uslova** za pojavu mrtve petje:

1. zauzimani **resursi se koriste u režimu međusobne isključivosti**,
2. **resursi se zauzimaju jedan za drugim**, tako da proces, nakon zauzimanja izvesnog broja resursa, mora da čeka da zauzme preostale resurse,
3. **resurse oslobađaju samo procesi koji su ih zauzeli**,
4. **postoji cirkularna međuzavisnost procesa** (prvi proces čeka oslobađanje resursa koga drži drugi proces, a on čeka oslobađanje resursa koga drži treći proces, i tako redom do poslednjeg procesa iz lanca procesa, koji čeka oslobađanje resursa koga drži prvi proces).

4. Kako se u praksi tretira problem mrtve petlje ?

Postoje **4 pristupa tretiranja** problema mrtve petlje:

1. **sprečavanje pojave mrtve petlje** (onemogućava važenje nekog od 4 uslova za njenu pojavu).
2. **izbegavanje pojave mrtve petlje**
3. **otkrivanje pojave mrtve petlje** i oporavak od nje
4. **ignorisanje pojave mrtve petlje**

5. Na čemu se temelji sprečavanje mrtve petlje ?

Kod sprečavanja pojave mrtve petlje:

- **važenje prvog uslova obično nije moguće sprečiti** –resursi se koriste u režimu međusobne isključivosti (jer se resursi najčešće koriste u režimu međusobne isključivosti).

- **važenje drugog uslova se može sprečiti** – resursi se zauzimaju jedan za drugim (ako se unapred zna koliko treba resursa i ako se oni svi zauzmu pre korišćenja).

- **važenje trećeg uslova se obično ne može sprečiti** – resurse oslobađaju samo procesi koji su ih zauzeli (jer najčešće ne postoji način da se zauzeti resursi privremeno oduzmu procesu).

- **važenje četvrtog uslova se može sprečiti –** postojanje cirkularne zavisnosti između procesa (ako se resursi uvek zauzimaju u unapred određenom redosledu, koji isključuje mogućnost cirkularne međuzavisnosti procesa).

6. Šta karakteriše izbegavanje mrtve petlje ?

Izbegavanje pojave mrtve petlje **zahteva poznavanje podataka o**:

-maksimalno mogućim zahtevima za resursima,

-ukupno postavljenim zahtevima za resursima

-stanju resursa

Praktična vrednost ovog pristupa nije velika, jer se obično unapred ne znaju maksimalno mogući zahtevi procesa za resursima. Sem toga, ova provera je komplikovana, a samim tim i neefikasna.

7. Šta karakteriše otkrivanje i oporavak od mrtve petlje ?

Proverava se **da li postoji proces**, **čijim se zahtevima ne može udovoljiti ni za jedan redosled zauzimanja i oslobađanja resursa.** Javlja se i problem, šta učiniti **kada se otkrije pojava mrtve petlje**. Ako se resursi ne mogu privremeno oduzeti od procesa, preostaje **jedino uništavanje procesa** radi oslobađanja resursa. Međutim, to nije uvek prihvatljivo i zbog toga ovaj pristup nema veliki praktični značaj.

8. Šta karakteriše ignorisanje mrtve petlje ?

Ignorisanje pojave mrtve petlje je **pristip koji se najčešće koristi u praksi**. Kada se mrtva petlja javi, **na korisniku je da se suoči sa ovim problemom i da ga reši** na način koji je primeren datim okolnostima.

lekcija 16

1. Od čega se sastoje komande znakovnog komandnog jezika?

Komande znakovog komandnog jezika se sastoje **od operatora i operanda**. Najjednostavniju komandu komandnog jezika predstavlja putanja izvršne datoteke.

2. Kako se zadaju komande grafičkih komandnih jezika?

Komande grafičkih komadnih jezika se zadaju **tako da se korisniku omogući da bira operator komande sa *spiska operatora* (menu).** Spisak operatora se prikazuje na ekranu, a izbor se vrši pomoću namenskih dirki ili miša. **Nakon izbora operatora sledi dijalog u kome korisnik navodi (ili opet bira) operand (operande) komande**.

3. Šta su ciljevi znakovnih komandnih jezika?

Ciljevi znakovnih komandnih jezika obuhvataju:

1) **omogućavanje izvršavanja pojedinih (korisničkih) programa,**
2) **omogućavaju kombinovanja izvršavanja više (korisničkih) programa,**
3) **omogućavaju pravljenja komandnih datoteka** (command file, shell script).

4. Šta omogućuju znakovni komandni jezici?

Omogućuju korisniku da zada komandu koja precizno određuje vrstu rukovanja i objekat rukovanja, a zadatak interpretiranja komande je da pokrene proces u okviru čije aktivnosti usledi rukovanje zatraženom komandom.

5. Šta omogućuju čarobni znaci?

Čarobni znakovi se mogu javiti u okviru operanada komandi, kao što je **znak ***. **Njegova upotreba je vezana za imena datoteka i namenjena je za skraćeno označavanje grupa datoteka.**
Zahvaljujući magičnim znakovima **moguće je jednom komandom uništiti sve objekte datoteke iz radnog imenika** (unisti*.obj), **ili odštampati sve tekst datoteke iz radnog imenika** (stampaj d*1.txt).

6. Šta omogućuje preusmeravanje?

Zahvaljujući obradi znakova moguće je interpreteru znakovnog komandnog jezika saopštiti i da *preusmeri* (redirect) standardni ulaz i standardni izlaz sa tastature i miša na proizvoljno odabrane datoteke. Ovo je važno za pozadinske procese koji nisu u interakciji sa korisnikom, pa **zahvaljujući preusmeravanju pozadinski proces ne ometa interaktivni rad korisnika.** Preusmeravanje standardnog ulaza najavljuje znak <, a preusmeravanje standardnog izlaza najavljuje znak >. **Preusmeravanje predstavlja osnovu za kombinovanje izvršavanja više korisničkih programa.**

7. Čemu služi *pipe*?

Umesto preusmeravanja standardnog ulaza i izlaza **moguće je nadovezati standardni izlaz jednog procesa na standardni ulaz drugog procesa i tako obrazovati *tok procesa* (pipe).**
Nadovezivanje u tok se označava pomoću znaka |.

8. Čemu služi baferovana specijalna datoteka?

Baferovana specijalna datoteka služi **za razmenu podataka između dva procesa koji su povezani u tok**. Ona služi prvom od procesa kao standardni izlaz, a drugom kao standardni ulaz. Prvi proces samo piše u ovu datoteku, a drugi samo čita iz nje.

9. Šta karakteriše pozadinske procese?

Pozadinski procesi se razlikuju od običnih (interaktivnih) procesa po tome što **interpreter znakovnog komandnog jezika nakon stvaranja pozadinskog procesa ne čeka kraj njegove aktivnosti nego nastavlja interakciju sa korisnikom.** Zato su pozadinski procesi u principu neinteraktivni.

10. Šta karakteriše komandne datoteke?

Komandne datoteke opisuju okolnosti pod kojima se izvršavaju korisnički programi, a sadržaj komandne datoteke preuzima na interpretiranje interpreter znakovnog komandnog jezika. Zato komandne datoteke imaju poseban tip da bi ih interpreter znakovnog komandnog jezika mogao prepoznati.

11. Šta omogućuju korisničke komande?

Korisničke komande omogućuju:

1) ***rukovanje datotekama*** (izmena imena datoteke, poređenje sadržaja datoteke, kopiranje datoteka, uništavanje datoteka)
2) ***rukovanje imenicima*** (komanda za stavranje i uništavanje imenika, za promenu radnog imena, za pregledanje sadržaja imenika i komanda za izmenu imena i ostalih atributa imenika),
3) ***rukovanje procesima,***
4) ***razmenu poruka između korisnika***.

12. Šta omogućuju administratorske komande?

Administratorske komande omogućuju:

1) **pokretanje i zaustavljanje rada računara**,
2) **spašavanje** (backup) **i vraćanje** (restore) **datoteka**,
3) **rukovanje vremenom**,
4) **sabijanje** (compaction) **datoteka**,
5) **ažuriranje podataka o korisnicima računara i njihovim pravima**,
6) **generisanje izveštaja o korišćenju računara**
7) **rukovanje konfiguracijom računara**,
8) **proveru ispravnosti rada računara** i
9) **pripremu diskova za korišćenje**.

lekcija 17

1. Šta karakteriše operativne sisteme realnog vremena?

Operativni sistemi realnog vremena su **namenjeni za primene računara u kojima je neophodno obezbediti reakciju na spoljašnji događaj u unapred zadatom vremenu**. Za operativne sisteme realnog vremena je **tipično da su zajedno sa računarom ugrađeni (embedded) u sistem**, čije ponašanje se ili samo prati ili čijim ponašanjem se upravlja. **Zadatak** operativnih sistema realnog vremena je **da samo stvore okruženje za korisničke programe**.

2. Šta karakteriše multiprocesorske operativne sisteme?

Multiprocesorski operativni sistemi **upravljaju računarskim sistemom sa više procesora opšte namene koji pristupaju zajedničkoj radnoj memoriji**. Podrazumeva se da su procesori i radna memorija povezani sabirnicom. Specifičnosti mikroprocesorskih operativnih sistema su vezane za modul za rukovanje procesorom i posledica su istovremene aktivnosti više procesa na raznim procesorima.

3. Koje module sadrži mikrokernel?

Hijerarhijska struktura mikrokernela:

1) **modul za rukovanje procesima,**
2) **modul za razmenu poruka,**
3) **modul za rukovanje radnom memorijom,**
4) **modul za rukovanje kontrolerima,**
5) **modul za rukovanje procesorom.**

4. Šta karakteriše poziv udaljene operacije (RPC)?

Poziv udaljene operacije (Remote Procedure Call) se vrši ako pozivana operacija ne odgovara potprogramu koji se izvršava u okviru aktivnog procesa pozivaoca, nego odgovara potprogramu koji se izvršava u okviru aktivnosti drugog, udaljenog procesa. Proces koji poziva udaljenu operaciju se nalazi u ulozi klijenta (primaoca usluge), a proces koji obavlja udaljenu operaciju se nalazi u ulozi servera (davaoca usluge). RPC **ima oblik potprograma** u kome se navode oznake(ime) operacije i njeni argumenti.

5. Šta radi klijentski potprogram?

Klijentski potprogram (client stub) **je potprogram koji poziva udaljene operacije**. Klijentski potprogram **obavlja skriveni niz koraka radi dobijanja zahtevane usluge**. U te korake spadaju:

-pronalaženje procesa servera koji pruža zahtevanu uslugu,

-pakovanje argumenata u poruku zahteva,

-slanje serveru ove poruke zahteva,

-prijem od servera poruke odgovora sa rezultatom pružanja zahtevane usluge,

-raspakivanje prispele poruke odgovora,

-isporuka rezultata pružanja zahtevane usluge pozivaocu klijentskog potprograma.

6. Za šta su zaduženi serverski potprogrami?

Postoje **dva serverska potprograma** (server stub) koje poziva jedino server i koji se obavljaju radi pružanja zahtevane usluge.

Prvi od serverskih programa **obuhvata**:

- **prijem poruke zahteva i**
- **raspakivanje argumenata iz ove poruke**.

Drugi serverski program **obuhvata:**

- **pakovanje rezultata usluge u poruku odgovora i**
- **slanje klijentu ove poruke odgovora**.

7. Koji problemi su vezani za poziv udaljene operacije?

Problemi koji se mogu javiti prilikom poziva udaljene operacije su:

1) **da se ne pronađe server koji pruža zahtevanu uslugu,**
2) **da se u toku prenosa izgubi ili poruka zahteva ili poruka odgovora**
3) **da dođe do otkaza ili servera ili klijenta u toku njihovog rada**.

8. Šta podrazumeva dinamičko linkovanje klijenta i servera?

Dinamičko linkovanje **podrazumeva pružanje zahtevane usluge u slučaju kada ima više servera iste vrste koji tada posebnom *serveru imena* ostavljaju podatke o sebi**. Serveru imena se obraćaju klijenti radi pronalaženja servera koji pruža zahtevanu uslugu.

9. Koje operacije podržava protokol razmene poruka između klijenta i servera?

Protokol razmene poruka između klijenta i servera **podržava sistemske operacije zahtevanja usluge i sistemske operacije prijema zahteva i slanja odgovora**.

10. Za šta su zadužene sistemske operacije koje ostvaruju protokol razmene poruka?

Te tri sistemske operacije su zadužene **za prenos poruka. Pored slanja i prijema poruka one potvrđuju prijem poruka, retransmituju poruke, šalju upravljačke poruke i slično**. U nadležnosti ovih sistemskih operacija je i rastavljanje i sastavljanje poruka u pakete.

11. Šta sadrže poruke koje razmenjuju klijent i server?

Svaka poruka koja se razmenjuje između procesa se sastoji od:

1. ***upravljačkog dela poruke*** (adresa odredišnog procesa, adresa izvorišnog procesa, opis poruke),
2. ***sadržaja poruke***.

12. Šta je potrebno za sigurnu razmenu poruka između klijenta i servera?

Za sigurnu komunikaciju između dva procesa je **potrebno da se oni obrate posebnom serveru, povereniku, kako bi dobili javni ključ svog komunikacionog partnera**. Za komunikaciju sa poverenikom procesi koriste unapred dogovoreni javni ključ poverenika, a za komunikaciju sa njima poverenik koristi njihove unapred dogovorene javne ključeve.

13. Šta karakteriše digitalni potpis?

Digitalni potpis se šalje uz poruku. On **sadrži podatke koji jednoznačno reprezentuju poruke, pa predstavlja *otisak prsta poruke*.** Otisak prsta poruke formiraju jednosmerne funkcije na osnovu sadržaja poruke. Digitalni potpis **nastaje kada se otisak prsta poruke dekriptuje** (transformiše) primenom algoritma dekriptovanja i privatnog ključa.

14. Od čega zavisi propusnost servera?

Propusnost servera **zavisi od njegove aktivnosti**. Ukoliko server ima strogo sekvencijalnu aktivnost to smanjuje njegovu propusnost i usporava pružanje usluga. Zato je za server potrebno obezbediti više niti.

15. Šta sadrže dozvole na kojima se zasniva zaštita datoteka u distribuiranom sistemu?

Dozvole sadrže:

1) **redni broj servera,**
2) **redni broj deskriptora datoteke,**
3) **oznaku vrste usluge,**
4) **oznaku ispravnosti dozvole.**

16. Šta karakteriše distribuiranu sinhronizaciju?

Distribuirana sinhronizacija **se ostvaruje razmenom poruka**. Sinhronizacija se može ostvariti na dva načina. Najjednostavniji način je **uvođenjem *procesa koordinatora*** kojim se **ostvaruje centralizovani algoritam sinhronizacije.** Potoje i **distribuirani algoritmi sinhronizacije** koji se zasnivaju na međusobnom dogovaranju procesa zainteresovanih za saradnju.

17. Šta karakteriše distribuirani računarski sistem?

Distribuirani računarski sistem je zamišljen tako da **integriše mnoštvo računara u moćan multiračunarski sistem.** Takav multiračunarski sistem **nudi veću pouzdanost kao i mogućnost proširenja**. Pored integrisanja pojedinačnih računara distribuirani računarski sistem **omogućuje i deljenje skupih resursa između više korisnika, prilagodljivost zahtevima korisnika, željenu raspoloživost, predvidivost odziva i veću sigurnost.**

18. Šta karakterise distribuiranu softversku platformu?

Distribuirana platforma (middleware) **ima ulogu dostribuiranog operativnog sistema**. Ona je obično specijalizovana tako da **nudi konzistentan skup operacija koje omogućuju razvoj željene vrste distribuiranih softverskih sistema.** U osnovi ovakvih sistema se krije klijent-server model.