# OPERACIJSKI SUSTAVI

**ZADACI ZA VJEŽBU – 3. CIKLUS**

(by kate)

# 8. Gospodarenje spremnickim prostorom

## 78. Gdje se generiraju adrese unutar procesora?

Programsko brojilo 🡪 adrese instrukcija

Registar kazaljke stoga 🡪 stogovne adrese

Sadržaj adresnih dijelova instrukcija 🡪 adrese podataka

## 79. Kako je podijeljen procesni adresni prostor?

Dretveni prostori + zajednički prostor

## 80. Opisati organizaciju smještaja sadržaja na magnetskom disku (cilindri, staze, sektori).

Disk se sastoji od više ploča na istoj osovini. Kružnice na ploči predstavljaju staze, a staze su podjeljene na jednake djelove (kružne lukove s jednakim središnjim kutem) – sektore. Staze jednakih polumjera svih diskova čine cilindar.

## 81. Cime je odredena jedinstvena adresa svakog sektora na disku?

* Rednim brojem ploče
* Rednim brojem staze na ploči
* Rednim brojem sektora na stazi

Trajanje traženja staze - trecina vremena potrebnog za prijelaz svih staza)

+ trajanje prijenosa podataka - ovisi o brzini prijenosa (br sektora\*br bajtova u sektoru/trajanje jednog okretaja) i kolicini prenesenih podataka

## 83. Od cega se sastoji trajanje traženja staze (*seek time*)?

* Ubrzavanje (ručice glave)
* Gibanje konstantnom brzinom
* Usporavanje
* Fino pozicioniranje

## 84. Koliko iznosi prosjecno vrijeme traženja u odnosu na vrijeme koje je potrebno za prijelaz preko svih staza?

Za vrijeme trazenja se uzima trečina vremena potrebnoh za prijelaz svih staza

## 82. Koliko iznosi ukupno trajanje prijenosa podataka tvrdi disk - radni spremnik? Pogledaj 85.

## 85. Kako je podijeljeno vrijeme koje je potrebno za prijenos podataka s diska ili na disk?

1. Trajanje postavljanja glave (head positioning time)

* Trajanje traženja staze (seek time)
* Ubrzavanje ručice glave
* Gibanje konstantnom brzinom
* usporavanje
* fino pozicioniranje
* rotacijsko kašnjenje ()

1. Trajanje prijenosa podataka (data transfer time)

* Trajanje čitanja dijela ili cijele staze

## 86. Zbog čega nastaje rotacijsko kašnjenje i koliko ono iznosi?

Rotacijsko kašnjenje nastaje zbog toga što se nakon postavljanja glave na odabranu stazu mora prije početka prijenosa pričekati da se ispod glave za čitanje pojavi traženi sektor.

Iznosi između nula(najpovoljniji slučaj traženi sektor upravo nailazi ispod glave u trenutku njezinog postavljanja na stazu) i

jednog (najlošiji slučaj  traženi sektor upravo prolazi ispod glave kad je ona postavljena na stazu) okretaja. Zato se za rotacijsko kašnjenje uzima prosječna vrijednost, odnosno pola trajanja jednog okretaja.

## 87. Čime je odredena brzina prijenosa podataka s diska u spremnik diskovne upravljacke jedinke?

Brzinom kojom ispod glave promiču bajtovi sektora.

## 88. Neka je trajanje traženja staze *Ts* nekog diska s 2000 staza opisano sa sljedece tri

## formule, gdje je *D* udaljenost izmedu trenutnog položaja glave i tražene staze:

## *Ts = 1.5* x *D ms za D* *4,*

## *Ts = 4.0 + 0.5* x *D1/2 ms za 4<D* *400,*

## *Ts = 10.0 + 0.01* x *D ms za D > 400.*

## Koliko iznosi prosjecno vrijeme traženja staze?

## 89. Navesti sadržaj procesnog informacijskog bloka.

Sve informacije važne za odvijanje procesa, sve informacije povezane s gospodarenjem  spremničkim prostorom, tablicu prevođenja, opisnike svih dretvi, opisnik virtualnog adresnog prostora. Ima ih koliko i procesa.

## 90. Opisati postupke statickog i dinamickog dodjeljivanja spremnika.

Statičko dodjeljivanje:

Particije su stalne veličine (fixed partitions), program kad je dodjeljen jednoj particiji uvijek je u toj particiji, i kad je izbačen iz radnog spremnika može se vratiti samo u tu particiju. Dolazi do problema fragmentacije i toga što adresni prostor programa nije mogao biti veći od najveće particije fizičkog spremnika.

*Dinamičko dodjeljivanje:*

Programi se upisuju jedan iza drugog. Dodan je sklop s kojim se početna adresa pribraja svakoj adresi koju proces proizvede. Programi se priređuju tako da svaki ima svoj adresni prostor. Važno je da se prije izvođenja u bazni registar pohrani aktualna početna adresa.

## 91. Navesti vrste fragmentacije prilikom statickog dodjeljivanja spremnika.

Unutarnja fragmentacija: Programi nisu jednake velicine kao particije pa ce djelovi particija ostati neiskorišteni.

Vanjska fragmentacija: Tijekom rada se može dogoditi da svi procesi čiji su programi smješteni u istu particiju bivaju blokirani pa ta particija radnog spremnika ostaje prazna. Pritom može postojati više procesa čiji programi čekaju na dodjelu radnog spremnika, ali oni ne mogu biti napunjeni u radni spremnik jer nisu pripremljeni za tu particiju.

## 92. Problem fragmentacije prilikom dinamickog dodjeljivanja spremnika se ne može izbjeci, ali se može ublažiti. Kako?

Rupe se održavaju što večima kako bi se u njih mogo smjestiti novi program.

* Pri svakom oslobađanju nekog procesnog prostora novo nastala rupa spaja se s eventualnim susjednim rupama u novu veću rupu
* Pri svakom novom zahtjevu za spremničkim prostorom potraži se najmanja rupa u koju se može smjestiti novi program

## 93. Unatoč tome što se problem fragmentacije prilikom dinamickog dodjeljivanja spremnika može ublažiti, fragmentacija može postati prevelika. Što treba tada učiniti?

Privremeno obustaviti izvođenje dretvi i „presložiti“ programe u kompaktni prostor.

## 94. Navesti i dokazati Knuthovo 50% pravilo.

Knuthovo pravilo: u stacionarnom stanju će u spremniku biti broj rupa jednak polovici broja punih blokova.

Izvod:

Knuthovo pravilo pretpostavlja da je vjerojatnost oslobađaja spremničkog prostora jednaka vjerojatnosti zahtjeva. Pretpostavlja

se da će zbog toga tijekom vremena u sustavu doći do stohastičkog ravnotežnog stanja odnosno da će prosječan broj rupa i

punih blokova biti stalan. U tom slučaju možemo predpostaviti da postoje 4 tipa blokova razvrstavjući ih po kriteriju da li im im jepredhodnik i sljedbenik rupa ili puni blok.

A xxx, B xx0, C 0xx, D 0x0

Veliko slovo je tip bloka, a malo slovo je broj blokova tog tipa. Blokova tipa A ima a.

Vjerojatnost da će se pronaći neka rupa koja će točno odgovarati veličini zahtjevanog prostora je zanemariva i označava se sa *q*, a vjerojatnost (*p= 1 - q*) je zato približno jednaka 1.

Sad idemo prebrojat rupe, blokove tipa ne brojimo jer oni nemaju rupa, a blokove tipa D brojimo dva puta jer imaju po dve rupe. E sad budući da ako, recimo, imamo blok tipa B(xx0) i blok tipa C(0xx) jedan do drugoga vidimo da smo ovu nulu brojili dva puta

 i zato moramo  cijelu sumu podjelit još sa dva.

**n(rupa) = (b+c+2d)/2**

Uz to budući da ako jednom bloku (C) slijedi nula, to mora značiti da će slijedećem (B)  prethoditi, što znači da će broj blokova tipa C i tipa B bit jednak**c=b**.

Znači,

**n(rupa)=(2b+2d)/2 = b + d**

**m(količina punih segmenata)=a + b + c + d**

Vjerojatnost da će nastati jedna rupa (znači mora nestat jedan blok tipa A) je **a/m**.

Vjerojatnost da će se popunit rupa (mora nestat blok tipa D) je **d/m + vjerojatnost** da postoji zahtjev veličine jedne rupe u koju će se novi program smjestiti (ili kraće q).

Zbog onog prvog uvjeta vjerojatnost popunjavanja i pražnjenja rupa moraju biti otprilike  jednaki znači:

**a/m = d/m + q ; pomnoži s m, zamjeni q sa 1‐p**

**a=d + m(1‐p)**

 uvrstimo sad taj a u

**m=a+b+c+d;**

**m=d + m(1‐p) +b+c+d ;**

 sad se sjetimo odprije da je c=b i b+d=n

**m=d + m‐pm + b +b + d**

**m=m‐pm +2(n)**

**pm = 2n**

**n=1/2pm ; udući da je p približno 1**

**n(rupa)=1/2m(memorije)**

 50% memorije čine rupe .

## 95. Kako treba podijeliti program (koji u cijelosti ne stane u radni spremnik) u prekloponom nacinu uporabe radnog spremnika?

Na jedan osnovni dio koji se uvijek nalazi u radnom spremniku i na dijelove koji se naizmjenično smještaju u preostali dio korisničkog dijela fizičkog spremnika.

## 96. Kako je podijeljen logicki, a kako fizicki adresni prostor u sustavu sa stranicenjem?

*Logički adresni prostor*se dijeli na jednako velike dijelove koje nazivamo **stranicama** (*page*). Prikladno je da stranice imaju veličinu koja je cjelobrojna potencija broja dva.

*Fizički adresni prostor* možemo podijeliti na dijelove koji su jednaki veličini stranice logičkog adresnog prostora. Te zamišljene dijelove nazivamo **okvirima** (*frames*). U jedan okvir fizičkog radnog spremnika može se smjestiti jedna stranica logičkog adresnog prostora

## 97. O cemu ovisi velicina fizickog i logickog adresnog prostora?

Logički adresni prostor ovisi o arhitekturi, a fizički o RAM-u.

## 98. Mogu li stranice logickog adresnog prostora biti smještene u okvire fizickog spremnika proizvoljnim redoslijedom?

Mogu. Time se izbjegava problem fragmentacije.

## 99. Cemu služi tablica prevodenja? Od kojih se elemenata sastoji?

Služi za prevođenje logičkog u fizički adresni prostor. Sastoji se od direktorija stranica i tablica stranica.

Tablica prevođenja ima dvije razine adresa organizirane po stranicama veličine 4 KB. U prvoj razini je jedna stranica s 1024 kazaljki od po 32 bita (direktorij stranica). Kazaljke direktorija pokazuju na stranice u kojima je opet smješteno po 1024 riječi od po 32 bita (tablica stranica)

## 100. U cemu se razlikuju prekidi izazvani zbog promašaja stranice kod stranicenja na zahtjev od ostalih vrsta prekida?

Prekid zbog adresiranja nepostojeće stranice se dogodi usred instrukcije. Nakon obrade prekida mora se ponoviti instrukcija unutar koje se dogodio prekid. Kod ostalih vrsta prekida, postojanje prekida se provjerava nakon izvodjenja instrukcije, a ponovno pokretanje dretve započinje sljedečom instrukcijom.

## 101. Opisati sklopovsku potporu za ostvarenje stranicenja u Intel x86 arhitekturi. Što sadrži i cemu služi TLB (Translation Lookaside Buffer)? Opisati tablicu prevodenja. Gdje se ona nalazi?

Intel nema registra povijesti. Stranice se razvrstavaju u dva razreda: one u koje se u prethodnoj periodi pisalo i one u koje se nije

 pisalo. Postoji jedan bit pristupa(A) i bit nečistoće(D). Prvo se pokuša izbaciti stranica kojoj je A=0. Inače se izbacuje stranica

kojoj je bit  nečistoće 0, odnosno čista stranica.

TLB je 32 bitni priručni međuspremnik za prevođenje adresa koji nam služi za cjelokupno spremanje fizičke adrese i dijela logičke adres, tj. omogućuje nam da ne moramo uzastopno dohvaćati fizičku adresu ako ju trebamo češće.

## Koliko puta je potrebno pristupiti radnom spremniku ako se stranica ne nalazi u TLB meduspremniku? 3 a koliko ako se nalazi u TLB medusprenmiku? 1

## 102. Koju informaciju nosi bit cistoce? Gdje se on nalazi?

Označava da li se sadržaj stranice mijenjao od njenog prebacivanja u radni spremnik. Ako se mijenjao kod izbacivanja iz radnog spremnika se mora prepisivati na disk, a ako nije onda se samo obriše iz radnog spremnika.

## 103. Čemu služi posmačni registar povijesti?

Za izvedbu LRU strategije. Pri punjenju stranice u neki odvir inicijaliziraju se na 0 njezin bit pristupa i posmačni registar. Kada se pristupi stranici bit pristupa stranice se postavlja u 1. Prekid od sata periodno posmakne sve bitove pristupa u posmačne registre i zatim bit pristipa izbriše. Sadržaj registra može se pročitati kao binarni broj. Manji broj pokazuje da se stranicu nije duže vrijeme koristilo. Kada se pojavi potreba za izbacivanjem stranice izbacuje se ona s najmanjim brojem.

## 104. Opisati sljedece strategije za izbacivanje stranica: FIFO, LRU, OPT te satni algoritam.

* FIFO – izbacuje se stranica koja je najdulje u radnom spremniku (first-in first-out)
* LRU – izbacuje se stranica koja se najdulje u proslosti nije upotrebljavala (least recently used)
* OPT (Optimalna strategija) – izbacuje se stranica koja se najdulje u budučnosti neće upotrebljavati
* Satni algoritam – Stranice se svrstavaju u listu po redu prispjeća. Lista se obilazi kružno posebnom kazaljkom (kada dođe do kraja kazaljka se vraća na početak liste). Kada se pojavi potreba za praznim okvirom izbacit će se stranica na koju pokazuje kazaljka ako je njezina zastavica A u 0, inače će se preskočiti i pritom promjeniti vrijednost A u 0. Kazaljka se pomiče za jedno mjesto dok ne naiđe na stranicu s A=0.

## 105. Opisati strategiju izbacivanja stranica u Intel x86 arhitekturi kada se u obzir uzimaju dvije zastavice za oznacavanje stanja stranice.

Zastavice koje se koriste su bit pristupa (A) i bit nečistoće (D). Prije se izbacuju stranice u kojima se nije pristupalo i koje su čiste.

## 106. U kojim stanjima se mogu nalaziti pojedini okviri tijekom rada?

* Aktivno stajne – okvir je dodjeljen jednom od procesa i njegov se redni broj (kazaljka koja pokazuje na njegov početak) nalazi u tablici prevođenja tog procesa
* Slobodno stanje – okvir se nalazi u povezanoj listi oslobođenih okvira
* Slobodno stanje s obrisanim sadržajem – okvir je spreman za dodjeljivanje

# 9. Datotecni podsustav

## 107. Navesti sadržaj opisnika datoteke.

* Naziv datoteke
* Tip
* Lozinka
* Ime vlasnika
* Prava pristupa
* Vrijeme stvaranja
* Vrijeme zadnje uporabe
* Ime posljednjeg korisnika, i slično
* Opis smještaja

## 108. Gdje su pohranjeni

## a) opis smještaja datoteke - u opisniku datoteke

## b) opisnik datoteke - na disku u nekom volume-u

## c) datotecna tablica - u opisniku datoteke

## 109. Navesti sadržaj datotecne tablice.

* broj sektora po disku
* broj slobodnih sektora
* informacije o slobodnim sektorima
* tablica opisnika pohranjenih datoteka

## 110. Na koji nacin se može prikazati slobodan prostor na disku.

* bitovni prikaz
* lista slobodnih sektora
* lista nakupina sektora

## 111. Opisati bitovni prikaz slobodnog prostora na disku.

U nizu bitova svakom sektoru odnosno nakupini sektora pripada jedan čija je vrijednost jednaka 1 kada je sektor odnosno nakupina zauzeta, inace 0.

## 112. Opisati prikaz slobodnog prostora u obliku liste slobodnih blokova.

Svaki element liste sadrzi tri lokacije. Prvo je kazaljka na sljedeci element, drugo adresa prvog slobodnog sektora, a trece broj slobodnih sektora.

## 113. Opisati nacin smještaja datoteka u UNIX datotecnim podsustavima (*i-node*).

Opisnik datoteke u UNIXu zove se i-node. Uz pretpostavku da je sektor velik 1KB u jedan sektor se može smjestiti 256 kazaljki. U opisniku se nalazi 13 kazaljki i to:

* 10 neposrednih kazaljki
* 1 jednostruko indirektna kazaljka
* 1 dvostruko indirektna kazaljka
* 1 trostuko indirektna kazaljka

Pristup do datoteke obavlja se na sljedeći način:

* Prvih deset sektora dohvaća se neposrednim kazaljkama čime se može dohvatiti sve sektore datoteka manjih od 10KB
* Jedanaesta kazaljka pokazuje na sektor u kojem se nalazi sljedečih 256 kazaljki (256 + 10)KB
* Dvanaesta kazaljka pokazuje na sektor u kojem su kazaljke na 256 sektora svaki s 256 kazaljki (256\*256+256+10)KB
* Trinaesta kazaljka pokazuje na trostuko stablo kazaljki (256\*256\*256+256\*256+256+10)KB = 16GB

## 114. Opisati nacin smještaja datoteka u NTFS datotecnom podsustavu. Što je MFT? Kako se pohranjuju „male“ datoteke?

Diskovni prostor dodjeljuje se po skupinama sektora (po 1, 2, 4 ili 8 sektora). NTFS sadrži datoteku MFT (glavna tablica datoteka)

u kojoj se nalaze opisnici svih datoteka (ukljucujuci i nje same). Jedan MFT zapis ima veličinu jedne skupine sektora (npr. 4KB). Za veče datoteke zapis sadrži indekse skupina sektora i ako je potrebno dodatno proširenje opisa.

Datoteka se dijeli na djelove koji su jednako veliki kao nakpine sektora (virtualne skupine). Za smještaj datoteka pronalazi se što više uzastopnih skupina sektora i dodjeljuje se datoteci (ako nema uzastopnih dodjeljuju se pojedinačne skupine.

Male datoteke smještaju se unutar MFT zapisa.