

eman ta zabal zazu



Universidad Euskal Herriko
del País Vasco Unibertsitatea
INFORMATIKA FAKULTATEA

2007/2008 ikasturtea

Sistema Eragileak I

Konputagailuen Arkitektura eta Teknologia Saila
Dpto. de Arquitectura y Tecnología de Computadores

2008ko ekainaren 3a

ARIKETAK

1. ariketa [1 puntu]

1. Bedi honako scripta, **ez_dakit** izenekoa.

```
for i in $*
do
  mkdir /tmp/$i
  for j in *
  do
    if [ -d $j ]
    then
      for k in $j/$i*
      do
        if [ -f $k ]
        then
          cp $k /tmp/$i
        fi
      done
    fi
  done
  rmdir /tmp/$i
done
```

- a) Zer egiten du **ez_dakit** ?
- b) Kodetu ezazu **ba_dakit** scripta, honakoa egingo duena: argumentu bezala katalogo baten izena eta testu kate bat jasoko ditu. Katalogo horretako sarreretatik pasatako testu kateaz hasten direnekin honakoa egingo du: fitxategi arrunta bada, irteera estandarrean bere lehen 20 lerroak idatziko ditu; azpikatalogoa bada, bere edukia pantailaratuko du; azkenik, ez bada ez fitxategi arrunta ez azpikatalogoa, irteera estandarrean "BESTELAKOA: " eta sarreraren izena idatziko du. Erabilpen adibidea:

```
ba_dakit /etc arik
```

2. ariketa [2 puntu]

DIFakidetza ospitaleak **historialak.dat** izeneko fitxategi bat dauka UNIX makina batean. Fitxategi honetan ospitalearen paziente guztien historialak gordetzen dira. Paziente bakoitzaren informazioa honako egitura du: lehen 64 byteak pazientearen izena eta abizenak dira, eta ondoren historiala bera dator, bere tamaina beti 32768 byte delarik (32 KByte).

64 byte	32768 byte	64 byte	32768 byte	...	64 byte	32768 byte
Paziente1	Historiala_paziente1	Paziente2	Historiala_paziente2	...	PazienteN	Historiala_pazienteN

UNIXeko Sarrera/Irteerako sistema-deiak erabiliz, kodetu ezazu C lengoaian bai irteera estandarrean eta bai **pazienteak.txt** fitxategian ospitaleko paziente guztien izena eta abizenak idatziko duen programa (paziente bat lerroko). Gerta daitezkeen errorearen kontrola egin behar da.

3. ariketa [1,5 puntu]

Interneteko atzipena eskaintzen duen enpresa batek bere bezeroen konexio guztiak LOG fitxategi baten gordetzen ditu. Fitxategiaren egitura honako da:

```
bezeroa    konexio_data    iraupena_seg
```

LOG fitxategiak bezeroek egindako konexio beste lerro izango ditu, konexioaren dataren arabera sailkatuta (yymmdd:hhmmss).

fakturatzaile izeneko programa bat dugu, azaldutako egitura duen fitxategi bat argumentu bakar bezala jasotzen duena. Programa honek, LOG fitxategiaren lerro bakoitzeko, irteera estandarrean lerro bat idazten du konexioaren kostuarekin (Euro ehunekotan). Adibidez, LOG fitxategia honakoa bada:

```
ane        080503:113010    120
koldo     080507:064504    3000
ane        080515:200019    660
kepa      080521:095054    48
```

fakturatzaile programak irteera estandarrean honakoa idatziko du:

```
ane        080503:113010    120    20
koldo     080507:064504    3000    500
ane        080515:200019    660    110
kepa      080521:095054    48      8
```

UNIXeko tresnez eta **fakturatzaile** programaz baliatuz, hornitzailearen fakturazioa testuzko fitxategi baten idatziko duen programa eraiki nahi da, bezeroaren izenaren arabera sailkatuta. Programa honen bi bertsio egitea eskatzen zaizu:

- Lehen bertsio bat UNIXeko **komando lerro bakar bat** erabiliz (suposatu LOG fitxategia **2008_maiatza.log** dela, eta sortu beharreko testuzko fitxategia **fakturazioa_2008_maiatza.txt** dela).
- Bigarren bertsio **multiprogramatu** bat C lengoaiatz, **fakturatzaile** programa eta UNIXeko **komandoak** erabiliz (ikus bukaerako zerrenda) eta beharrezko iruditzen zaizkizun izenik gabeko buzoiez (*pipe*-ak) baliatuz. LOG fitxategia eta sortu behar den emaitza fitxategia egin beharreko programaren argumentuak izango dira. Erabilpen adibidea:

```
nire_programa    LOG_fitx    emaitza_fitx
```

- Bezero bakar baten fakturazioa lortu nahiko bagenu (adibidez, **ane**), adieraz ezazu zertan aldatuko litzatekeen a) ataleko soluzioa.

4. ariketa [2,5 puntu]

DIF_BANK banketxeak bezeroentzako kontsulta zerbitzu berri bat jarri nahi du, Bezero/Zerbitzari eredian oinarrituta, eta zerbitzu hau UNIX makina batean martxan ipiniko da. Zerbitzu honek, bi eskaera mota desberdin onartuko ditu: azken hilabeteko mugimenduak, eta azken urteko mugimenduak. Horretarako, bezeroak *struct eskaera* egituran oinarrituta eskaera bidaliko du **MBX_DIF_BANK** zerbitzariaren buzoira. Bezeroa, zerbitzariaren erantzunaren zain geratuko da. Erantzunaren formatua *struct erantzuna* egituran oinarrituko da.

```
struct eskaera {
    char NAN[20];
    int eskaera_mota;
        // 0: azken hilabetea
        // 1: azken urtea
    char mugimendu_fitx[80];
    char bez_buzoia[20];
}
```

```
struct erantzuna {
    char NAN[20];
    int eskaera_onartua;
        // 1: onartua
        // 0: ez onartua
}
```

Zerbitzariak ziurtatu behar du eskaeran jasotako NAN zenbakia, benetan banketxearen bezero bat dela. Horretarako hurrengo **funtzioa** dugu (exekuzio azkarrekoa):

```
int benetako_bezeroa(char *NAN);
```

Funtzio honek 1 balioa itzuliko du bezeroa existitzen bada, eta 0 ezezkoaren kasuan. Bezeroa ez bada existitzen, momentuan erantzun behar zaio, erantzunaren eskaera_onartua atalean 0 balioa esleituz. Aldiz, bezeroa existitzen bada, eskaera zerbitzatzen amaitzean erantzungo zaio. Eskaera zerbitzatzeko bi aukera desberdin ditugu, eskaera mota kontuan izanda:

- Eskaera, azken hilabeteko mugimenduei buruz bada, hurrengo **funtzioa** dugu (exekuzio azkarrekoa):

```
int azken_hilabeteko_mugimenduak(char *NAN, char *mugimendu_fitx);
```

- Aldiz, eskaera azken urteko mugimenduei buruz bada, urte osoko informazioa BACKUP diskoetan gordeta dagoenez, hurrengo **programa** dugu:

```
azken_urteko_mugimenduak NAN mugimendu_fitx
```

Programa honek, argumentu bezala, bezeroaren NANA eta azken urteko mugimenduak idatzi nahi diren fitxategiaren izena jasoko ditu.

Azaldutako zerbitzua UNIX makina zaharkitu eta kaxkar batean ezartzea erabaki da (proba moduan). Horregatik, eskaera bakarra zerbitzatzeko da une bakoitzean.

Hurrengo eskatzen da:

1. Marraz ezazu eskema bat sistema osatuko duten prozesu guztiak azalduz eta elkarren arteko komunikazio mekanismoak argi adieraziz.
2. Idatz ezazu zerbitzariaren eta lagungarri ikusten dituzun beste prozesu guztien kodea.
3. DIF_BANK banketxeak UNIX makina berritzea erabaki du. Ondorioz, makina berri honetan, gehienez 100 eskaera zerbitzatu daitezke uneoro. Marraztu egoera berri honi dagokion eskema berria, azalduz egin beharreko aldaketak. Azaldu, ere, egin beharreko aldaketak kodearen atalari dagokionez.

Sistema-deiak:

```

int open(char *izena, int modua, int baimenak);
int close(int kanala);
int read(int kanala, char *buf, int luz);
int write(int kanala, char *buf, int luz);
int lseek(int kanala, long posiz, int nondik);
    // posiz: 0 hasiera, 1 unekoa, 2 bukaera
int link(char *iturburua, char *helburua);
int unlink(char *izena);
int stat(char *izena, struct stat *sbuf);
int fstat(int kanala, struct stat *sbuf);
int chdir(char *izena);
int fork();
int execlp(char *izena, char *arg0, char *arg1, ..., char *argn, NULL);
int execvp(char *izena, char *arg[]);
int wait(int *egoera);
void exit(int egoera);
unsigned long time(0);
int pipe(int *pfd);
int mkfifo(char *izena, int baimenak);
int mknod(char *izena, int baimenak, int unitatea);
int dup(int kanala);
DIR *opendir(char *path);
int closedir(DIR *dir);
struct dirent *readdir(DIR *dir);
    struct dirent { long d_ino;        // Inode zenbakia
                   char *d_name;     // Izena
                   }

```

Funtzioak:

```

char *strcpy(char *helburu, char *jatorri);
char *strcat(char *katea1, char *katea2);
int strcmp(char *katea1, char *katea2);
int strlen(char *katea);
char *strstr(char *katea, char *azpikatea);
int atoi(char *katea);
char *itoa(int zenbakia);
int sprintf(char *katea, char *formatua, ...);

```

UNIX-eko programa eta utilitateak:

```

chmod - fitxategien baimenak aldatzeko
echo - mezuak idazten ditu irteera estandarrean
grep - azpikateak bilatzen ditu fitxategietan
head - fitxategien lehen lerroak aukeratzen ditu
last - sistemara egindako konexioak pantailaratzen ditu
ps - prozesuei buruzko informazioa
sort - fitxategien lerroak alfabetikoki sailkatzen ditu
tail - fitxategien azken lerroak aukeratzen ditu
wc - lerroak, hitzak, eta karaktereak kontatzen ditu
who - sistemara konektatutako erabiltzaileei buruzko informazioa

```