

1. Praktika

1. atala: UNIX. Fitxategi-Sistema eta Komando-Interpretatzailea (KI)

2. atala: C lengoaiako programen konpilazioa gcc-rekin

Kontzeptuak

Izen absolutua eta erlatiboa, bidea (*path*), baimenak, jabea, loturak, dispositiboarekiko independentzia, berbideratzeak, pipeak, atzeko mailako atazak (*background*), filtroak.

Iturburu-kodea, konpilazioa, konpilazio banatua, objektu-kodea, estekatzea, fitxategi exekutagarria, karga, exekuzioa, erreferentzien ebazpena.

Enuntziatua

1. atala: UNIXeko fitxategi-sistema zuhaitz moduan dago antolatutik, bestelako loturak egin daitezkeelarik. Fitxategi sistema honen ezagutza eta maneia izango da lehen helburua: izenak, katalogoak, fitxategiak, nabigazioa sisteman zehar, loturak. Erabiltzaile anitzeko sistema denez, atzipen baimenak ere kudeatuko dira. Filtroen eta berbideratzeen bidez dispositiboarekiko independentzia ikusiko da. Komando-Interpretatzaileak eskaintzen duen hainbat aukera ere aztertuko dira.

2. atala: emandako adibideak erabiliz praktikatuko gcc konpiladorearekin.

Urratsak

1. atala: proposatutako ariketak egin, *bash* KI-rekin praktikatuz.

2. atala: proposatutako ariketak egin, *gcc* konpiladorearekin praktikatuz.

Dokumentazioa

- UNIXeko emandako apunteak.
- C lengoaiako ikastaroko apunteak.
- gcc konpiladorearen aukeren apunteak.
- C programen iturburu kodea.
- UNIXeko laguntza (*man*).

Linux-erako sarrera

Ezaugarriak

- + Sistema eragile multiprogramatua, erabiltzaile anitzekoa eta interaktiboa
- + Miniordenagailuetan eta lan-estazioetan estandarra
- + Komando-interpretatzaile ahaltsua
 - . berbideratzeak (< > >>)
 - . *pipe* komunikazioetarako (|)
 - . *background* edo atzeko mailako atazak (&)
- + Utilitate-programa asko
 - . aplikazio linguistikoak
 - . komunikazioak
 - . ezagutzaileak eta konpiladoreak sortzeko tresnak
 - . testu-prozesaketa
- Komandoak kriptikoak dira
- Informazio eta segurtasun gutxi sistematik

Komandoen sintaxia

Sinbolo bereziak:

```
<   berbideratu sarrera estandarra (adib: prog < sarr_fitx)
>   berbideratu irteera estandarra (adib: prog > irt_fitx)
>> berbideratu irteera baina eransketa-moduan (adib: prog >> era_fitx)
<< sarrera datuak ondoren datoz (script-etan erabiltzen da)
&   atzeko mailako ataza, ondoren datozenekin konkurrentzian (adib: prog &)
|   ataza konkurrenteak: 1.aren irteera 2.aren sarrera (adib: prog1 | prog2)

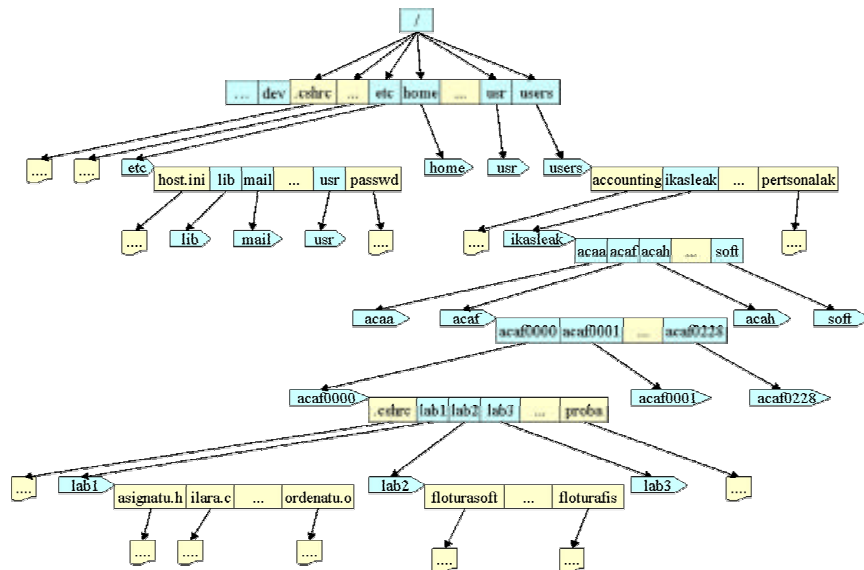
(Bakarrrik beraien sarrera edo/eta irteera estandarra erabiltzen duten programen sarrera
edo/eta irteera berbideratu daiteke. Era berean, pipe baten ezkerrean/eskuinean
programa bat jartzeko bere irteera/sarrera estandarra erabili behar du)
```

Ordezkapen karaktereak (metakarakterek):

- * edozein karaktere-kate ordezkatzen du, baita kate hutsa ere
- ? karaktere bakarra ordezkatzen du

Bi karaktere hauetako bat komando bateko hitz baten agertzen bada, *shell*-ak (*bash* gure kasuan) hitza patroia bezala tratatuko du, patroia betetzen duten katalogoko sarrera guztien zerrenda alfabetikoaz ordezkatuz.

Fitxategi-sistema



- Zuhaitz erakoa, erro bakarrarekin (/).
- Unitateen muntaketa/dismuntaketaren beharra
- Fitxategien izena = [bidea]/sarrera
 - bide absolutua (erroarekiko) /kat1/kat2/kat3/fitx
 - bide erlatiboa (uneko katalogoarekiko) fitx
kat3/fitx
../../kat2/kat3/fitx
./ uneko katalogoa
../ guraso katalogoa

Oinarrizko komandoak

- Shell-etik irteteko (lan saioa bukatzeko): **exit**, **logout**

exit

logout

Gure erabiltzaile kontuaren pasahitza aldatzeko: **yppasswd**

yppasswd

- Komandoei buruzko laguntza: **man**

man komandoa

Komando edo funtzio bati buruzko laguntza ateratzen du irteera estandarrean.

Adibidea:

man ls

- Katalogoak listatzeko: **ls**

ls [aukerak] [kat]

Adierazitako katalogoaren sarrerek ateratzen ditu irteera estandarrean.

AUKERAK (batzuk bakarrik)

- a sarrera guztiak, puntuz (.) hasten direnak barne
- l informazio osoa: baimenak, loturak, jabea, taldea, tamaina, data, izena
- R azpikatalogoak ere listatzen ditu era errekursiboan

Adibideak:

ls

ls -a

ls -al

ls -l kat1

- Uneko katalogoa aldatzeko: **cd**

cd [katalogoa]

Uneko katalogoa adierazitakora aldatzen du. Helburu katalogoaren izena bide absolutua (errotik) edota erlatiboa (uneko katalogotik) bidez adierazi daiteke. Ez bada katalogorik adierazten, erabiltzailearen jatorrizko katalogora aldatzen du.

- Uneko katalogoa zein den jakiteko: **pwd**

pwd

Uneko katalogoaren bide absolutua erakusten du irteera estandarrean.

- Katalogo bat sortzeko: **mkdir**

```
mkdir [bidea/]katalogoa
```

Bidea ez bada adierazten, uneko katalogoan sortzen du katalogo berria.

- Katalogo bat ezabatzeko: **rmdir**

```
rmdir [bidea/]katalogoa
```

Katalogoa ez badago hutsik, ez du ezabatzen eta errore mezu bat erakusten du.

- Irteera estandarrean argumentuekin lotutako testu-mezuak ikusteko: **echo**

```
echo arg1 arg2 ... argN
```

- Fitxategiak kopiaatzeko: **cp**

```
cp fitx_iturri fitx_helburu
```

```
cp fitx_iturri1 ... fitx_iturriN kat_helburu
```

```
cp -R kat_iturri kat_helburu
```

Fitxategi bat kopiaatzeko balio du. Katalogo bat kopiaatzeko ere balio du. Baita fitxategi bat edo gehiago beste katalogo batean kopiaatzeko.

- Fitxategiei izena aldatzeko: **mv**

```
mv iturri helburu
```

```
mv iturri1 ... iturriN kat_helburu
```

Fitxategi edota katalogo bati izena aldatzeko balio du. Azken argumentua existitzen den katalogoa bada, orduan aurreko fitxategi guztiak duten izenarekin katalogo horretara mugitzen ditu.

- Fitxategiak ezabatzeko: **rm**

```
rm sarreral ... sarreraN
```

```
rm -R kat
```

```
rm -i fitx
```

Argumentu bezala pasatako fitxategi guztiak ezabatzen ditu. Katalogo ez hutsak ez ditu ezabatzen (-R adierazten bada, orduan errekursiboki katalogoa eta bere azpiko guztia ezabatzen du). -i adierazten denean baieztapena eskatzen du.

- Irteera estandarrean egutegia erakutsi: **cal**

```
cal [hila [urtea]]
```

Uneko hila erakusten du. Hila edo/eta urtea adierazten denean, eskatutakoa erakusten du.

- Fitxategien eduki osoa ikusteko (baita zenbait fitxategi kateatzeko ere): **cat**

```
cat fitx_zerrenda
```

Pasatako fitxategiak irteera estandarrean erakusten ditu, kateatuz. Ez bada argumenturik pasatzen, sarrera estandarra erakusten du.

Adibideak:

```
cat fitx1.txt
```

```
cat fitx1.txt fitx2.txt
```

- Fitxategien edukia pantailaz pantaila ikusteko: **more**

```
more fitx
```

- Fitxategien hasierako lerroak ikusteko: **head**

```
head [-n] fitx
```

Ez bada beste kopururik adierazten, lehen 10 lerroak erakusten ditu.

Adibideak:

```
head fitx1.txt
```

```
head -18 fitx1.txt
```

- Fitxategien bukaerako lerroak ikusteko: **tail**

```
tail [-n] fitx
```

Ez bada beste kopururik adierazten, azken 10 lerroak erakusten ditu.

Adibideak:

```
tail fitx1.txt
```

```
tail -8 fitx1.txt
```

- Filtroak: **grep**

```
grep [-v] patroia fitx_zerrenda
```

Pasatako fitxategien edukia aztertzen du, eta patroia duten lerroak erakusten ditu irteera estandarrean. Ez bada fitxategirik pasatzen, sarrera estandarra hartzen du sarrera bezala. -v adierazten bada, patroia ez duten lerroak erakusten ditu.

- Loturak: **ln**

```
ln [-s] exist_izena izen_berria
```

Hardware lotura: -s adierazi gabe. Dagokion katalogoan sarrera berri bat sortzen du, existitzen den sarreraren inodeari apuntatzen diona (*inodearen zenbakiaren bidezko lotura*).

```
ln exist_izena izen_berria
```

Software lotura: -s adierazita. Dagokion katalogoan sarrera berri bat sortzen du LINK motakoa, bere edukia existitzen den sarreraren izena delarik (*fitxategiaren izenaren bidezko lotura*).

```
ln -s exist_izena fitx_berria
```

- Fitxategiak editatzeko: **pico**, **emacs**, **vi**...

```
pico testu_fitx
```

```
emacs testu_fitx
```

```
vi testu_fitx
```

- Fitxategien eta katalogoen baimenak aldatzeko: **chmod**

```
chmod modua fitx_zerrenda
```

Pasatako fitxategiei irakurketa (r), idazketa (w), edota exekuzio (x) baimenak aldatzen die. Aldaketa jabearentzat (u), taldearentzat (g), beste erabiltzaileentzat (o) edota guztientzat (a) egin daiteke. Baimen hauek 9 biteko segida bezala kodetarik azaltzen dira fitxategi bakoitzeko.

-rwxrwxrwx
Jabea Taldea Besteak

Modua argumentuak baimen berriak adierazten ditu, bi eratara eman daitezkeelarik: hiru zenbaki ortzitar bezala (4: irakurketa, 2: idazketa, 1: exekuzioa), edota karaktere kate bezala (u|g|o|a nori adierazteko, +/- baimenak gehitu ala kendu egiten diren adierazteko, eta r|w|x zein baimen ematen edo kentzen diren esateko).

Adibideak:

```
chmod 740 *.txt (rwx r-- ---)
```

```
chmod u+x fitx (jabeari 'x' ipini)
```

Beste zenbait komando

- Konektatuta dauden erabiltzaileak ikusi: **who**, **w**

- Prozesu aktiboak ikusi: **ps**

- Prozesu bat amaitu: **kill**

```
kill [-9] prozesu_zenbaki
```

- Fitxategiak inprimatzeko prestatzeko, orritan banatzeko, identifikatzeko, zutabeak definitzeko, etab.: **pr**

- Fitxategiak inprimatzeko, normalean modu atzeratuan (*spooling*): **lpr**

```
lpr -m - fitx1 (amaieran mezua)
```

```
lpr -Plaser fitx1 (laser izeneko inprimagailuan)
```

- Inprimatze ilara ikusteko: **lpq**

```
lpq -Plaser (laser izeneko inprimagailuan)
```

- Baimen estandarrek: **umask**

- Fitxategiaren jabea eta taldea aldatzeko: **chown** eta **chgrp**

- Fitxategiak bilatzeko fitxategi-sisteman: **find**

```
find / -name fitx -print
```

- Irteera lehenetsia bikoizteko (irteera ikusi eta gorde nahi denean): **tee**

- Lerroak, hitzak eta karaktereak kontatu: **wc**

- Fitxategien lerroak alfabetikoki sailkatu: **sort**

- Unix mota: *uname*
- Martxan daraman denbora: *uptime*
- Fitxategi-sistemaren erabileraren egoera: *df*

Adibideak

```
rm *.txt          txt motako fitxategi guztiak ezabatzen ditu
cat maiz?.txt      izena "maiz"ez hasi, ".txt"ez bukatu eta tartean
                   karaktere bakarra duten fitxategiak kateatu
lpr maiz[1-3].txt  maiz1, maiz2 eta maiz3 inprimatzen ditu
cat fitx1 >fitx2   cp fitx1 fitx2 egitearen baliokidea
head fitx1 >>buru  fitx1 fitxategiaren lehenengo 10 lerroak
                   metatzen ditu buru-ren bukaeran
man cp >lis1 &     cp-ren man orria kopiatzen du lis1
                   fitxategian, bitartean lanean jarrai daitekeelarik
head -50 fitx1 | lpr fitx1 fitxategiaren lehenengo 50 lerroak
                   inprimatzen ditu
```

Proposatutako ariketak

1. Irten sistematik eta berriro sartu
2. Aldatu zure pasahitza, eta egokiago bat ipini
3. Exekutatu hurrengo komandoak eta aztertu emaitza:


```
echo Kaixo      zemuz      zaude?
echo "Kaixo    " zemuz      zaude?
echo "Kaixo      zemuz      zaude?"
```
4. Hilabete honen egutegia atera pantailan.
5. Uneko katalogoaren izena lortu, bide absolutua adieraziz
6. Uneko katalogoaren edukia atera
7. Uneko katalogoaren eduki *guztia* atera
8. /users/alumnos katalogoaren edukia atera
9. Zure kontuaren *home* katalogoan kokatu eta aztertu ea hurrengo komandoak berdinak diren edo ez:


```
cd ../../../../usr
cd /usr
```
10. Zure lan-katalogoan sortu **Lab1** izeneko azpikatalogoa
11. Sortu berri duzun katalogoan kokatu (*Lab1* katalogoa) eta ondoren dagoen katalogo egitura sortu:


```
---|-- Jatorria -----|---- bin
      |                   |---- datuak -----|---- urtarrila
      |                   |                   |---- otsaila
      |                   |                   |---- martxoa
      |                   |---- lib
      |-- Gauzak
      |-- Zaborra
```
12. *datuak* katalogoan kokatu. Aztertu zein katalogotara joaten zaren hurrengo komandoak exekutatzera:


```
cd ../../
cd ../../Gauzak
cd ../Gauzak
cd urtarrila/../../Gauzak
cd urtarrila/../../Gauzak
cd /users/alumnos/acaf/acafxxxx/Lab1/Gauzak
cd $HOME/Lab1/Gauzak
cd
```
13. Zure kontuaren *home* katalogoan kokatu eta sortu software lotura bat (SOFT izeneko) hurrengo katalogoari: /users/alumnos/soft/acaf/eusk
14. Software lotura bidez sortu duzun katalogora mugitu

15. Lab1 katalogoan kokatu eta adibide.txt izena duen fitxategia sortu.
Fitxategiaren edukia, gutxienez, 40 lerrokoa izango da

16.adibide.txt fitxategia adibide1.txt izena duen fitxategira kopia

17.adibide1.txt fitxategia Gauzak azpikatalogoan kopia

18.adibide.txt fitxategia Gauzak azpikatalogoan kopia adibide2.txt izenarekin

19.Gauzak katalogoko adibide2.txt fitxategia adibide4.txt izenarekin berrizendatu

20.Lab1 katalogoan kokatu eta Gauzak2 izeneko azpikatalogoa sortu

21.Gauzak azpikatalogoko edukia Gauzak2 azpikatalogoan kopia

22. Lab1 katalogoan kokatu, hurrengo komandoa exekutatu eta aztertu bere eragina.
cp -R Gauzak Gauzak3

23.Ondorengo komandoak exekutatu eta aztertu emaitzak:
cat adibide1.txt
more adibide1.txt

24.Makinara konektatuta dauden erabiltzaileen zerrenda lortu.

25.Gauzak katalogoan kokatu eta lortu adibide1.txt fitxategiko lehenengo 10 ilarak.

26.Lortu adibide1.txt fitxategiko azkeneko 10 ilarak.

27.Lortu adibide1.txt fitxategiko azkeneko 4 ilarak.

28.Lortu adibide1.txt fitxategiko 20 eta 30 arteko ilarak.

29.adibide1.txt fitxategiko ilara kopurua zenbatu.

30.adibide1.txt fitxategiko hitz kopurua zenbatu.

31.adibide1.txt fitxategiko karaktere kopurua zenbatu.

32."kaixo" karaktere katea duten adibide1.txt fitxategiko ilarak lortu.

33."kaixo" karaktere katea EZ duten adibide1.txt fitxategiko ilarak lortu.

34.adibide1.txt fitxategia adibide3.txt izenarekin kopia.

35.adibide1.txt fitxategia adibide33.txt izenarekin kopia.

36.adibide1.txt fitxategia adibide333.txt izenarekin kopia.

37.Ondorengo komandoak exekutatu eta aztertu emaitzak:
ls adibide*
echo ls adibide*
ls adibide?.txt
echo ls adibide?.txt
echo adibide?.txt
echo adibide*

38.adibide333.txt fitxategia ezabatu.

39.Ondorengo komandoa exekutatu eta aztertu emaitza:
echo rm *

40.Gauzak azpikatalogoan kokatu, ondorengo komandoak exekutatu eta aztertu emaitzak:
ls -l
more adibide1.txt
chmod u-r adibide1.txt
ls -l
more adibide1.txt
chmod u+r adibide1.txt
ls -l
more adibide1.txt

41.Gauzak2 azpikatalogoari exekuzio baimena kendu.

42.Ondorengo komandoak exekutatu eta aztertu emaitzak:
ls -l
cd Gauzak2
ls Gauzak2
chmod u+x Gauzak2
ls -l
ls Gauzak2
cd Gauzak2

43.Ondorengo komandoak exekutatu eta aztertu emaitzak:
cat adibide1.txt
cat adibide1.txt > emaitza.txt
ls
cat emaitza.txt
ls > emaitza.txt
cat emaitza.txt

44.a1.txt fitxategia sortu eta hiru testu ilara sartu.

45.Ondorengo komandoak exekutatu eta aztertu emaitzak:
cat a1.txt
cat a1.txt > emaitza.txt
cat emaitza.txt
cat a1.txt >> emaitza.txt
cat emaitza.txt
ls >> emaitza.txt
cat emaitza.txt

gcc konpiladorea

Urratsak konpilazioan (banan-banan edota pausu bakarrean)

- Aurreprozesaketa:** aurreprozesadorearen aginduak (#define...) interpretatzen dira, iturburu fitxategietan behar diren ordezkapenak eginez. Adibidez:

```
more adibide.c
```

```
#define TAMAINA 50
main(int argc, char *argv[]) {
    long taula[TAMAINA];
    int i;

    for (i = 0; i < TAMAINA; i++) {
        taula[i] = i*i;
        printf("%d balioaren karratua %d da\n", i, taula[i]);
    }
}
```

```
gcc -E adibide.c > adibide.pp
```

```
more adibide.pp
```

```
main(int argc, char *argv[]) {
    long taula[50];
    int i;

    for (i = 0; i < 50; i++) {
        taula[i] = i*i;
        printf("%d balioaren karratua %d da\n", i, taula[i]);
    }
}
```

- Konpilazioa:** C iturburu kodea makinaren prozesadorearen mihiztadura lengoaiara itzultzen da.
- Mihiztadura:** mihiztadura kodea objektu kodera itzultzen da (kode bitarra, prozesadoreagatik exekutagarria dena).
- Estekatzea:** programa osatzen duten objektu modulu desberdinak (iturburu fitxategiak konpilatuak, erabilitako liburutegiak...) estekatu egiten dira, fitxategi exekutagarri bakarra sortuz.

gcc konpiladorearen sintaxia

gcc opzioak C_fitx_izena obj_modulu_izena

Opzioak:

- c** objektu fitxategia soilik lortzeko (estekatu gabe)
- g** araztatzailearentzako (*debugger*) informazioa sortzeko
- llib_izena** estekatzaileari (*linker*) esaten dio lib_izena liburutegian objektu moduluak bilatzeko esaten dio
- Lkatalogoa** liburutegiak dauden katalogoa adierazteko
- o exek_izena** lortzen den exekutagarriari izena emateko
- E** aurreprozesatzea soilik egiteko
- S** mihiztadura kodea soilik lortzeko
- D** sinboloak definitzeko (baldintzako konpilazioa, makroak)

Adibideak

gcc adibide.c (emaitza: a.out)

gcc -c adibide.c

gcc adibide.o -o adibide

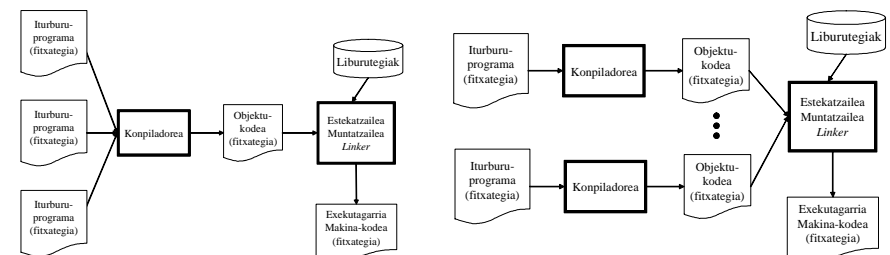
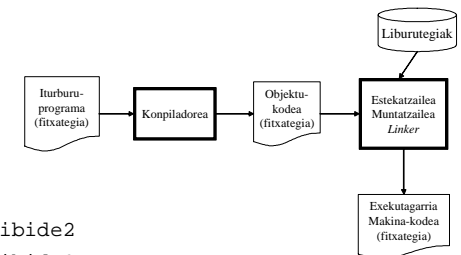
gcc adibide.c -o adibide

gcc -c adibide2.c

gcc -c errutinak.c

gcc adibide2.o errutinak.o -o adibide2

gcc adibide2.c errutinak.c -o adibide2



ld tresna (*link-editor*)

Moduluak estekatzen ditu

ld opzioak obj_modulu_izenak

Opzioak:

-lliburutegia objektu-moduluak aurkitzeko liburutegia
-Lkatalogoa liburutegiari dagokion katalogoa
-o exek_izena lortzen den exekutagarriari izena emateko
-M moduluen mapa lortzen da

ar tresna

Objektu-moduluen liburutegi bat sortu eta eguneratzen du.

ar opzioak liburutegia obj_modulu_izenak

Opzioak:

d osagairen bat ezabatzeko
m osagairen bat bukaeran jartzeko
p osagairen bat inprimatzeko
r osagairen bat ordezkatzeko
t edukien taula listatzeko
x osagairen bat ateratzeko
c liburutegia sortzeko
u aldatuak baino ez ordezkatzeko
v aldaketei buruzko informazioa lortzeko

Liburutegiak

(1) Izena zera izan behar da: libizen.a
Adibidez: libmod.a

(2) Sorkuntzarako: ar rcv libizen.a
Adibidez: ar rcv libmod.a

(3) Objektu modulu-bat gehitzeko: ar r libizen.a modulu.o
Adibidez: ar r libmod.a cambiar.o

(4) Konpilazioa liburutegia erabiliz:
gcc -Lkat -o exekutagarria iturburua -lizen
Adibidez: gcc -L. -o proba proba.c -l**mod**
libmod.a liburutegian bilatzen dira objektu-moduluak

(5) Gauza bera estekatzailetik:
ld -Lkat -lizen -o exekutagarria objektua.o
Adibidez: ld -L. -l**mod** -o proba proba.o

gdb tresna (arazketa)

Sintaxia: **gdb** exekutagarria

Aginduak:

help	aginduen laburpena
run	exekutatu programa
next	exekutatu agindu bat
step	funtzioaren barruan sartu
list	kodea ikusi
break	eten-puntu bat definitu
print	aldagai baten balioa azaldu orain
display	aldagai baten balioa azaldu eten-aldi guztietan
where	exekuzio-puntua eta funtzioen pila ikusi

Enuntziatua

1. ariketa. Taulen kudeaketarako C lengoian egindako funtzio multzo bat (*taulak.c*) ematen da. Funtzioak erabiliz zenbakiak sailkatzeko programa nagusi bat programatu, konpilatu eta probatu.

2. ariketa. Ilaren kudeaketarako C lengoian egindako funtzio multzo bat (*ilarak.c*) ematen da. Objektu-modulua sortuko da multzo horretatik. Beste bi fitxategi ematen dira ere (*ilarak.h* eta *asignatu.h*) funtzioen erazagupena ez errepikatzeko eta memoriaren kudeaketa egiteko. *proba.c* programa —eginda ematen dena— zenbaki osokoak irakurri, ilararatu eta idazten ditu. Programa honetan oinarriturik sailkatzeko programa bat eraiki beharko da.

Urratsak

1. sailkatu.c programa idatzi, taulak.h sartuz (#include agindua).
2. sailkatu.c fitxategia konpilatu, sailkatu programa sortuz. Probatu.
3. ilarak.c fitxategia konpilatu, ilarak.o sortuz.
4. proba programa sortu probatu.
5. Aldatu proba zenbakiak sailkatu ditzan.

```
/* taulak.h */

extern void irakur_taula(long taula[], int kop);
extern void inprimatu_taula(long taula[], int kop);
extern void sailkatu(long taula[], int kop);
```

```
/* taulak.c */

#include <stdio.h>

void irakur_taula(long taula[], int kop)
{
    int i;

    printf("Sakatu %d zenbaki:\n", kop);
    for (i = 0; i < kop; i++)
        scanf("%ld", &taula[i]);
}

void inprimatu_taula(long taula[], int kop)
{
    int i;

    printf("Taularen edukina:");
    for (i = 0; i < kop; i++)
```

```
        printf(" %ld", taula[i]);
        printf("\n");
    }

int txikiena(long taula[], int kop)
{
    int i, ind_txik = 0;

    for (i = 1; i < kop; i++)
        if (taula[i] < taula[ind_txik]) ind_txik = i;
    return(ind_txik);
}

void sailkatu(long taula[], int kop)
{
    int i, ind;
    long tnp;

    for (i = 0; i < kop-1; i++) {
        ind = txikiena(&taula[i], kop-i);
        tnp = taula[i];
        taula[i] = taula[i+ind];
        taula[i+ind] = tnp;
    }
}
```

```
/* ilarak.h */

struct item {
    struct item *ondoko;
    short lehen;
    char *infor;
};

struct ilara {
    struct item *lehen;
    struct item *azkena;
};

#define NIL (struct item*)0
#define BOOL short

void sortu(struct ilara *p);
BOOL huts(struct ilara *p);
struct item *lehen;
struct item *atera(struct ilara *p, short leh);
void lotu(struct ilara *p, struct item *pelem);
void txertatu(struct ilara *p, struct item *pelem);
```

```
/* asignatu.h */

#define MAXELEM 10

struct item taula_elem[MAXELEM];

struct item *hartu_elementu()
```

```

{
    static int i = 0;

    if (i < MAXELEM) return(&taula_elem[i++]);
    else return(NIL);
}

```

```

/* ilarak.c */

#include "ilarak.h"

void sortu(struct ilara *p)
{
    p->lehena = NIL;
    p->azkena = NIL;
}

BOOL huts(struct ilara *p)
{
    return(p->lehena == NIL);
}

struct item *lehena(struct ilara *p)
{
    struct item *paux;

    paux = p->lehena;
    if (paux != NIL) {
        p->lehena = paux->ondoko;
        if (p->azkena == paux)
            p->azkena = NIL;
    }
    return(paux);
}

void lotu(struct ilara *p, struct item *pelem)
{
    struct item *paux;

    paux = p->azkena;
    if (paux == NIL)
        p->lehena = pelem;
    else
        paux->ondoko = pelem;
    pelem->ondoko = NIL;
    p->azkena = pelem;
}

struct item *atera(struct ilara *p, short leh)
{
    struct item *paux, *paur;

    paux = NIL;

    for (paux = p->lehena; paux != NIL; paux = paux->ondoko) {
        if (paux->lehen > leh) paux = paux;
        if (paux->lehen == leh) {
            if (paur != NIL) paux->ondoko = paux->ondoko;

```

```

            else p->lehena = paux->ondoko;
            if (paux->ondoko == NIL) p->azkena = paux;
            return(paux);
        }
        if(paux->lehen < leh) return(NIL);
    }
    return(NIL);
}

void txertatu(struct ilara *p, struct item *pelem)
{
    struct item *paux, *paur;

    paux = NIL;

    for (paux = p->lehena; paux != NIL; paux = paux->ondoko) {
        if (pelem->lehen <= paux->lehen) paux = paux;
        else break;
    }
    pelem->ondoko = paux;
    if (paur != NIL) paux->ondoko = pelem;
    else p->lehena = pelem;
    if (paux == NIL) p->azkena = pelem;
}

```

```

/* proba.c */

#include <stdio.h>
#include "ilarak.h"
#include "asignatu.h"

struct ilara ilara_bat;

main()
{
    short datu;
    struct item *elem;

    sortu(&ilara_bat);
    while (scanf("%hd", &datu) != EOF) {
        if ((elem = hartu_elementu()) == NIL) break;
        elem->lehen = datu;
        lotu(&ilara_bat, elem);
    }
    printf("\n");
    while (!huts(&ilara_bat)) {
        elem = lehena(&ilara_bat);
        printf("%hd\n", elem->lehen);
    }
}

```