

# Sistema Eragileen oinarriak - 1. azterketa partziala

## 1. Linux programazioanako tresnak eta hauen erabilera.

► GALDETEGIA: Linuxekiko hasierako harremana.

1. "man komandoa" idatziz terminalaren komandoa zertarako den eta nola erabili azalduko zaizue. man komandoaren laguntzaz lotu itzazu agindua bere zereginarekin.

.. → aurreko direktorioa

pwd → uneko direktorioa zein den ezagutzeko.

less fitxategia → more komandoaren antzekoa.

cat fitxategia → "fitxategia"ren edukia ikustarazteko.

cd dir → "dir" direktoriara aldatzeko.

nano → testu-editore ez grafikoa.

more fitxategia → "fitxategia"ren edukia orri orri ikustarazteko.

logout → terminalaren uneko saioa amaitzea.

passwd → pasahitza aldatzea

man komandoa → "komandoa"ren erabilera ezagutzeko.

► KODEA: 1Adibidea.c

// Mezuak adierazten zaien adina adiz errepikatzen ditu

```
#include <stdio.h>
#include <stdlib.h>

int main (int argc, const char *argv[])
{
    int i,j;
    if (argc < 3) {
        printf("Erabilera: %s errepikapenak mezuak\n", argv[0]);
        exit(1);
    }
    if (atoi(argv[1]) < 0)
        exit(1);
    for (i=1; i < atoi(argv[1]); i++)
        for (j=2; j < argc; j++)
            printf("[%d] %s\n", i, argv[j]);
    exit(0);
}
```

• int main (int argc, const char \*argv[])

• exit(1) vs. exit(0) → exit(1): amaiera arrakastatsua; exit(0): errore amaitza.

• %s → string motako argumentu bat } printf  
%d → int motako argumentu bat }

• atoi → int atoi(const char \*str) string argumentua int bihurtzen du.

• argv → argumentu kopurua, exekutagarriaren izena lehen argumentua beti.  
argv → argumentuen bektorea, bektorean argumentuak gordetzen dira.

## 2. Sistementzat utilitateen bateragarritasuna

### ► GALDETEGIA : Windows /Linux bateragarritasuna

1. Linux-en erabiltzeko kompilatuta programa exekutagarria arazorik gabe abiarazi daitake Windows-en, beti ere bi sistema eragileek makina berdina partekatzen badute.

↳

2. Linux-en erabiltzeko kompilatutako programa exekutagarria ezin da Windows-en abiarazi, baina iturburu-kodea Windows-erako berriro kompilatu dezakegu C-ko kompildore batekin eta ondoren arazorik gabe abiaraz daitake.

↳

3. Linux-en erabiltzeko kompilatutako programa aldatu eta berriro C-ko kompildore batekin eta ondoren arazorik gabe abiaraz daitake.

↳

### ► Iratzargaiwa. c - Windows-erako bertsioa

// Denbora tarte bat itxoin eta gero mezu bat idatzten du.

```
#include <stdio.h>
#include <stdlib.h>
#include <windows.h>

unsigned itxoin_denbora (int s) {
    if (s<=0) return 0;
    else {
        Sleep(s * 1000);
        return s;
    }
}

int main (int argc, const char * argv []) {
    int i;
    if (argc < 3)
        printf ("Erabilera: %s denbora mezua \n", argv[0]);
        exit(1);
    itxoin_denbora (atoi(argv[1]));
    for (i=2; i< argc; i++)
        printf ("%s \n", argv[i]);
}
```

► KoDEA: Iratzargaiwa.c - Bertso bateragarria

// Windows-erako zein Linux-erako balio du

```
#include <stdio.h>
#include <stdlib.h>
#ifndef WIN32
#include <windows.h>
#else
#include <unistd.h>
#define Sleep(x) usleep((x)*1000)
#endif
```

```
unsigned itxo_in_denbora (int s){
```

```
    if (s <= 0)
        return 0;
    else
        Sleep (s*1000);
    return (s);
}
```

```
int main (int argc, const char * argv [] ) {
```

```
    int i;
    if (argc < 3)
        printf ("Erabilera: %s denbora mezua\n", argv[0]);
        exit(1);
}
```

```
    itxo_in_denbora (atoi (argv[1]));
    for (i = 2; i < argc; i++)
        printf ("%s\n", argv[i]);
}
```

## 1. Sistemas Eragileak ► DEFINIZIOAK

- Komandoak: terminalean hitz espezifiko bat sartzean exekutatzen diren programa txikiak dira, eragiketa jakin bat burutzeko. Hitz horiek jadanik definiria dawdu.
- Scriptak: Komandoz osaturiko testuzko fitxategiak dira, komando lerroan exekutatu daitezkeenak.
- Sistema Eragilea: erabiltzaileak hardwarearekin komunikatzeko bidea da. Aplikazio eta hardwarearen artean zubi bat bezala funtzionatzen du.
- Liburategia: programa baten zati bat bereiztea, berrerabiltsako programari pisua kentzeko.
- Bateragarritasuna: programa bat eta sistema, aplikazio edo arkitektura bat elkar uzta zuzen utetzeko baldintza da.
- Liburutegi-errutinak: liburutegi batek publikoki eskaintzen dituen errutinak dira.
- Sistema-deiak: sistema-eragileak eskaintzen dituen errutinak babestutako eremuarekin lan egiteko.
- Trapak: programa baten kodean egin daitezkeen erroreak sortzen dituzten etenak dira, "exception" ere deitua.

## 3. Sistema Eragileen Historia

### 3.1. Sistema Eragile Klasikoak

- Zer dira Sistema Eragileak?  
Sistemas eragileak programen exekuzioa kudeatzeko sortu ziren, informatikaren lehen urteetan (1950eko hamarkadan).
- Batch sistemak:  
Aurreneko sistemak batch sistemak ziren. Erabiltzaileak prozesadoreak exekutatu beharreko programa txartel berezi batzuetan (batch txarteletan) "programatzen" zuen; hauetan zuloak eginez. Gero irakurgailu baten bidez irakurtzen ziren, informazioa prozesadoreari igorriaz. Batch bakoitzean txertatuta zeuden kontrol-txartelak monitorearen aginduekin.
- Multiprogramazio edo Multi-ataza sistemak:  
Sarral irteera gailuak prozesadorea baino motagoak ziren batch sistemetan; eta ondorioz prozesadorea lanik gelditzen zen askotan. Ingeniarik arazo honi aurre egiteko nonakoa egin zuten: Sistema eragileentzat eta hainbat programentzat memoria nahikoa badago, prozesadorea S/I bati itxaroten ari den exekuzio edo lan bati itxaroten ari den exekuzio edo lan bati "kasu egiten" egin beharrean; S/I bati itxaroten ari EZ den beste lan bati kasu egitea.

## ► Sistema elkareragileak

Erabiltzailearekin harremana daukaten sistemak. Sistema honek erabiltzaileak egindakoaren arabera eginkizun bat edo beste egingo du. Hau da, pertsona batek eta makina batek elkarri eragiten dioten sistema bat.

## ► Denbora partekatutako sistemak

Sistema honek multiprogramazio teknikak eta erloju kontrolak erabiltzen ditu sistemaren baliabideak (puz, memoria...) erabiltzaile guztien artean banatzeko. Horrela denbora ondo aprobatxatzen da erabiltzaile batek informazioa bidaltzen ez duen denbora tartean beste batek bidaliko bantu informazioa. Exekuzio horren garaien pertsonak ezin du parte hartu.

## ► Erabiltzaile amitzeko sistemak

Sistema horren programak bi erabiltzailek edo gehiagok batera erabiltzea aukera ematen duen sistema. Erabiltzaile hauen kudeaketa hasieran sistema operadoreak esku  $\Rightarrow$  egin behar zuen, baina gaur egun sistema hauek erabiltzaileak kudeatzeko mekanismo automatikoak dituzte.

## 3.2. Konputagailu pertsonalen sistema eragileak

### ► Erabiltzaileak

1970ko hamarkada amaineran  $\rightarrow$  hardwarea merkatu eta mikropozesadoreak agertu ziren. Konputagailu pertsonalak erabiltzaile batarrekoak, eta ataza bakarrekoak eta babes-mekanismorik gabeak. Terminalen jardunez, 1990ko hamarkadatik aurrera  $\rightarrow$  konputagailu pertsonalen hardwareak multi-ataza sistemak onartzen zituen (macOS, Windows, Linux) eta erabiltzailearen interfaze grafiko aurreratuvak ere onartzen dituzte. Gaur egungo konputagailu pertsonalak erabiltzaile amitzekoak dira; eta ondorioz erabiltzaileak sistemaren administratzalea izan behar du.

### ► Sistema autonomoak

Ordura arte, enpresa handien multiprogramatutako sistema zentralak urrunetik, terminal "tontoetatik", atzi zitezkeen. Konputagailu pertsonalen terminal hauek ordiezkatu dituzte eta orain sistema autonomoak dira, hauek konputazio-ataza batzuk burutu ditzaketen sistema autonomoak dira eta sistema zentralera deskarga dezakete. Konputagailu pertsonalak beraien artean ere komunikatu daitezke; gaur egun beraien artean sare-azpiegitura baten bidez komunikatzen den konputagailu-multzo bat ematen da.

### ► Bezeroa $\leftrightarrow$ zerbitzaria

Baliabide zehatz batera sarbidea eskaintzen eta kudeatzen duen sareko makina izango da baliabide horren zerbitzaria. Bezeroek baliabidea atzitzen dute bezero-zerbitzari eskema baten bidez. Konputagailu-sareen agerpen eta hedapenak asko korapilatu du ez balkarrik sistema eragilea, baita ere bertan implementatzen diren zerbitzuak (middleware). Interneteko esparruan hedatzen diren sistema banatuak sortu dira eta zerbitzuuen eskema eta kontzeptu oso elaboratuak sortu dituzte: web-zerbitzuak eta cloud computing zerbitzuak.

## ► Linux

Linux (1991), UNIX sistema-familiako kidea da eta konputagailu pertsonaletan erabiltzena zuzenduta dago, eta software libreko produktu bat da. Software librearen ezaugarni nagusia da ez duela kopia eta banaketa mugariak, jabedun softwareak ez bezala. GNU bezalako erikundeeek software librearen kopia, eraldaketa eta birbanaketa lizenziak ematen dituzten baldintza bakarrarekin, distribuazio berriak iturburu-kodea barnetitu behar du.

## 3.- Gailur mugikor eta txertatuentzako sistemak

### ► Mugikortasuna

Hardwarearen garapena ez da konputagailu pertsonaletan amaitzen, geroz eta txikiagoak eta eraginkorrakoak dira. Konputagailuak eramangarririk bihurtu dira: bateriekin, eta kable gabeko komunikazioak erabiliz.

### ► Nonahiko Sistema

Gailuak sare bihur daitezke eta beraien artean erlazionatu daitezke, ondorioz portaera adimendua izan litekeena eskainiz.

Hauetako nonahiko sistema deitzen zaie eta ingurune-adimenen kontzeptua erabiltzen da ingurune honetan sortutako aplikazioak izendatzeko. Etorri zunean agertuko dira gailu orain txikiagoak, txartel adimenduak eta (mota) sentsoreak bezaldu; hauetan ingurune fisikoko edozein puntuko informazioa prozesatu eta ematen dute.

### ► Denbora errealeko sistemak

Sistema eragile mota espezifikoak garatzen joan dira, nabarmenduz denbora errealeko sistemak. Askotan sistema mota hauetan sistema konplexuagotan barneratu daude. Denbora errealeko sistemetan erantzunak epemuga bat bete behar du. Epea gainditu bada, erantzuna ez da baliozkoa eta katastrofikoa izan daiteke. Horregatik garatu dira denbora errealeko sistema eragile espezifikoak. Erabilera orokorreko sistema eragileak ere denbora errealeko atazekin bateragarririk dira, baina egokiak dira bakarrik epea ez betetzea kritikoa ez denean.

### ► Sistema eragileen ibilbide ziklikoa

Historian zehar, hardware euskarri berrietara moldatzeko, sistema eragileek egin duten ibilbide ziklikoa nabarmendu behar da.

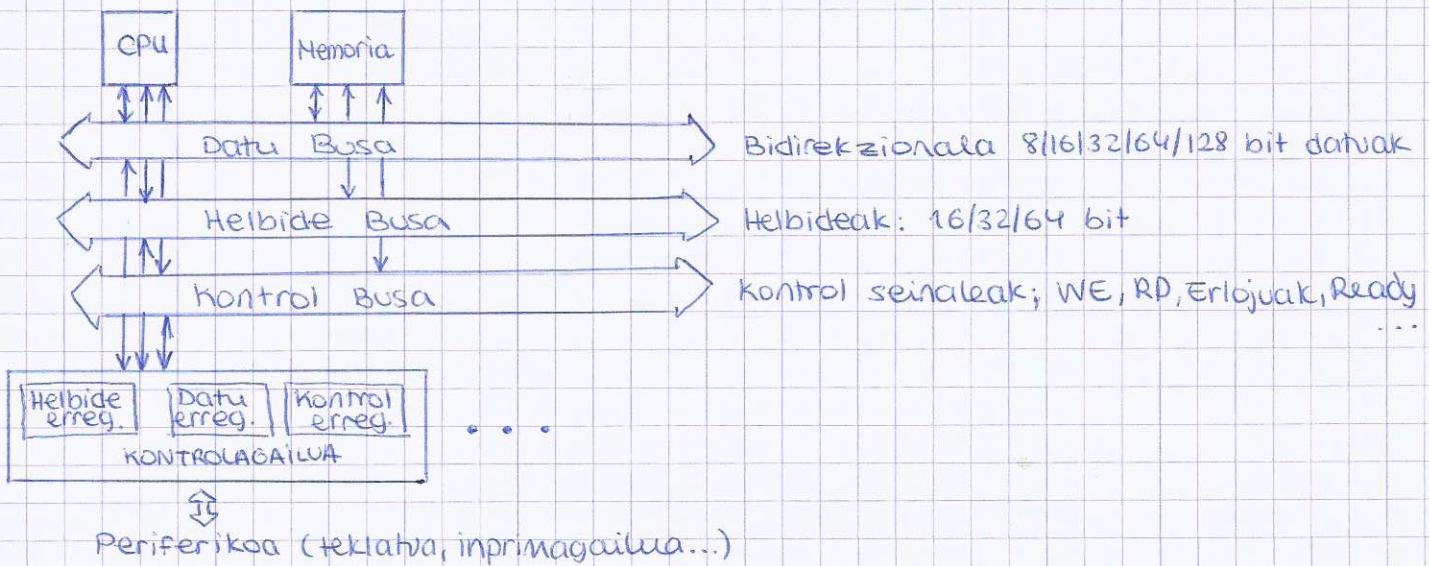
Konputagailu pertsonalen hardwarea garatu denean, beren sistema eragileek teknika hauetan barneratu dituzte eta gaur egun oso konplexuak dira. Miniatirizazioak tamaina txikiko eta prestazio murri zutoño gailu mugikor berrien agerpena bultzatu da azkenaldian. Ondorioz, historia errepikatu egiten da: telefono mugikorren aurreneko sistema eragileak oso soila ziren, baina gaur egungo bertsioak jada multi-ataza dira. Txartel adimenduek eta mote-holozio bera jarraituko dute?

• Mota (sentsoreak): ingurune fisikoko edozein puntuko informazioa prozesatu eta ematen dute.

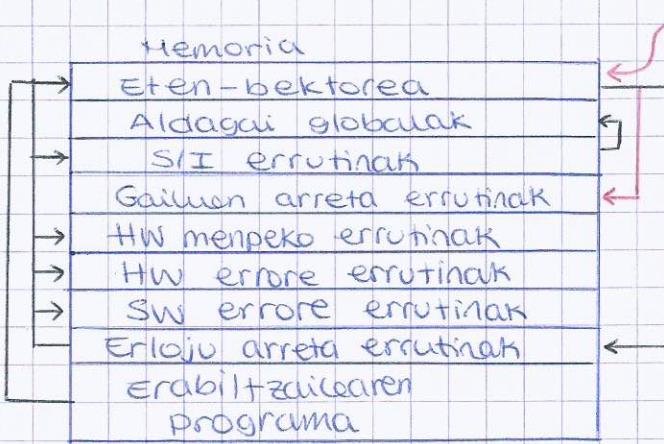
4. Sistema-deiak eta hauen araberako programen eta sistema eragileen egitura.

### Sistema-deien eskema

→ Von Neumann arkitektura



→ S/Iren eskema orokorra (etenen bidezko)

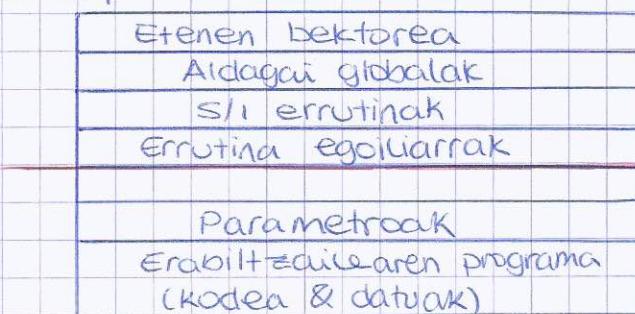


### Etenen bidezko mekanismoa

- Mekanismoaren deskribapenak:
- Programatzailaren aldetik makinaren ezagutza.
  - HW dependentziak, aldaketak?
  - S/I errutinen egeneraketak
  - Erabiltzailearen programen mantenketa, eta bere bizi-zikoa.

→ Errutina egoiliarrak sistema eragilearen hastapen gisa

### Memoria mapa



### Eten mekanismoa

### Sistema Eragilea

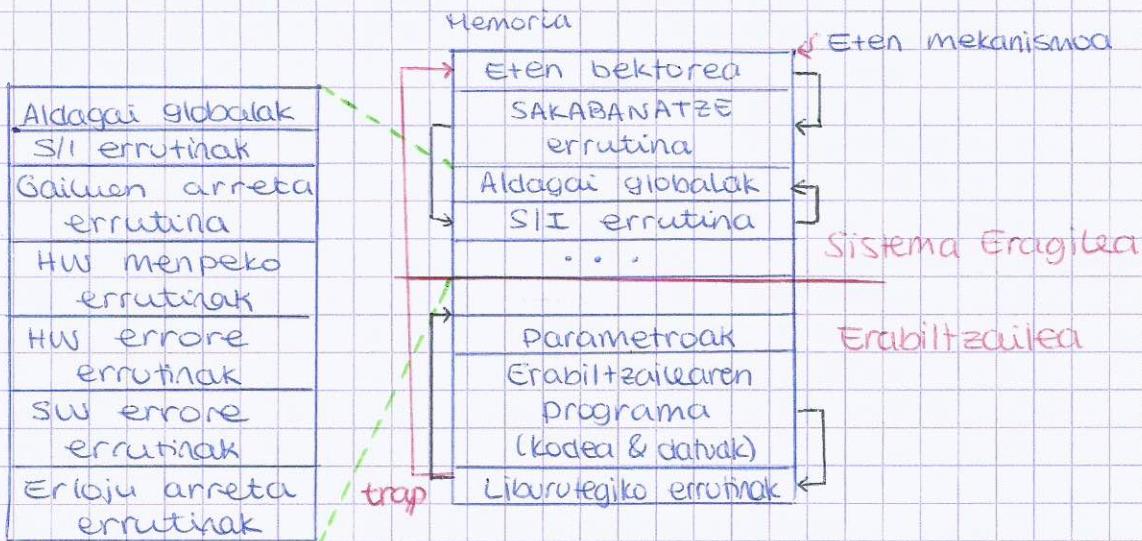
### Erabiltzailea

Egoiliar (INFORMATIKA): Ordenadore bateko sistema eragilearen edota programa gainbegiratzailearen zati den programaz edo komandoz esaten da, memoria zentralean beti eskura dagoena, egikaritzeko (exekutatzeko) prest.

[Iturria: Harluxet Hiztegi Entziklopedikoa]

→ Atzipen eredu bateratua

↳ Eten bektorearentzako beharrezko tamaina murriztu



## 5. Sarrera / Irteera eta errore kanalak

### Shell-eko komandoak

- echo
- cat
- ls
- sort
- cp
- < eragigaila

## 6. Iragazkiak

- Nola portatzen da iragazki bat?

Linuxen bi komando elkarlotu daitzke, lehenengoaren irteerak bigarrenaren sarrera gisa joka dezan. Horretarako | simboloa erabiltzen da. komando batek beste baten irteera berorren sarrera gisa hartzeari, honen gain operazio bat exekutatzeari, eta emaitza sarrera estandarrean inprimatzeari iragazkia deritzogu.

ADB grep komandoak parametro gisa jasotzen du expresio erregular bat, horren arabera inprimatz sarrera estandarrean emaitza. sort komandoak, ordena alfanumerikoaren arabera ordenatzen du, sarrera estandarrean inprimatz.

- Zer sistema-dei erabiltzen ditu eta nolakoak dira parametroak?

• `ssize_t Read(int fd, const void *buf, size_t count)`: sarrera estandarretik irakurriko du sartutako datua. Horretarako `Read(0, *buf, BUFFER-TAMAÑO)` idatzi behar dugu, 0 delako sarrera estandarren parametroa. Dei honek `ssize_t` motako datua itzuliko du.  
• `Write(int fd, const void *buf, size_t count)`: irteera estandarretik irakurriko du sartutako datua. Horretarako `Write(1, *buf, BUFFER-TAMAÑO)` idatzi behar dugu, 1 delako irteera estandarren parametroa.

- Nola adierazten dira S/I estandar kanalak?

IZENA	FITXATEGIAREN IDENTIFIKADOREA	LEHENETSITAKO HELAIDEA
Sarrera Estandarra (stdin)	0	tekiatura
Irteera Estandarra (stdout)	1	pantaila
Errore Estandarria (stderr)	2	pantaila

`stdin / stdout / stderr` → liburutegi-errutinak

0 / 1 / 2 → sistema-deiak

## 7. Fitxategien atzipen ez sekuentziala.

Liburutegi errutinak buffer bat erabiltzen dute, sistema eragilea libratzen denean sistema deia erabiltzen dute bufferreko informazio hori sistema eragileak kudeatutako dezan eta gorde. Hori sistema eragilea ez gainkargatzeko egiten da.

Sistema deien ordez bitartez egiten denak ordea, konputagailuko baliabide guztiak erabiltu ditu. Eta sistema eragilea da informazioa kudeatzen duena (etengabe), arazoren bat gertatzen bada informazio hori gordeta izango duelarik.

## SISTEMA ERASIGLE KLASIKOAK

Programen execuzioa kudeatzeko (komputagailu, handi, garesti, mugatu)

- Batch-sistemak → aurrez grabatutako eta zulotutako txapel-multzoak prozesatzeko.
- Batchen bidezko seqüentiacioa ~~SI~~ SI gailua prozesadorea baino moldeago.
  - ↳ Prozesadoreak SI gailuren oinarrizko jasotzean, eragiketa amaitu arte, itxaron.
  - ↳ Prozesamendu - estrategia eraginkorrak garatu
- Multiprogramatza → Prozesadoreak SI batz i txaroten dira ez den beste lan batetara auka dezake eta horrela lanpetuta mantenteen da.
- Elkarreragilea → terminalaren aurkem jarruteko eragiketa moduluek garatu. Erantzun - denbora laburra.
  - ↳ Prozesadorea etengabe lanpetuta mantendu dezake
- Denbora partekatutako sistemak →
- UNIX → jabetzen sistemen sifurra hautsi.
  - ↳ UNIX C-z berridatzi eta arkitektura guzhei eramangarría.
- Erabiltzaile amitzeko komputagailuak

Sistema-deiak: programari irakur erabilizten duen mekanismoa sistema eragileari zerbitzu batz eskatzeko.

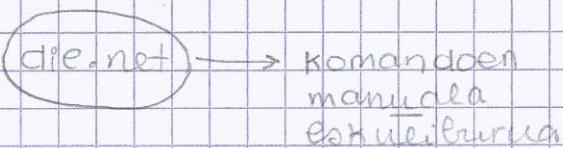
Errutina egilarrak: komputagailu bateko SEREN edota programa griaibegiratzailukoen zati den programaz edo komandoz eseten da, memoria centralaean beti eskura dagoena, egitarazteko prest. (executatzeko).

→ Harlauzet hiztegi entziklopedikoa

Librutegia: programaz osatutako fitxategi multzoa.

Sistema-deiak ↔ Sistema bakar bixitza bat.

iratzargailuq.c



```
//////////  
// 1Adibidea.c //  
// Mezuak adierazten zaion adina aldiz errepikatzen ditu //  
//////////  
  
#include <stdio.h>  
#include <stdlib.h>  
  
int main (int argc, const char * argv[])  
{  
    int i, j;  
  
    if (argc < 3) {  
        printf("Erabilera: %s errepikapenak  
mezua\n", argv[0]);  
        exit(1);  
    }  
    if (atoi(argv[1]) < 0)  
        exit(1);  
    for (i=1; i<=atoi(argv[1]); i++)  
        for (j=2; j<argc; j++)  
            printf("[%d] %s\n", i, argv[j]);  
    exit(0);  
}
```

```
//////////  
// Iratzargailua.c  
// Denbora tarte bat itxoin eta gero mezua bat idazten du //  
//////////  
  
#include <stdio.h>  
#include <stdlib.h>  
#include <unistd.h>  
int main (int argc, const char * argv[]){  
    int i, j;  
  
    if (argc < 2) {  
        printf("Erabilera: %s itxarote-denbora mezua\n",  
               argv[0]);  
        exit(1);  
    }  
    sleep(atoi(argv[1]));  
    if(argc == 1){  
        printf("Defektuzko mezua.\n");  
    }  
    else  
        printf("%s\n",argv[2]);  
}
```

```
////////// Iratzargailu_bateragarria.c //////////////////////////////////////////////////////////////////
// Iratzargailu_bateragarria.c // Denbora tarte bat itxoin eta gero mezu bat idazten du // Denbora tarte bat itxoin eta gero mezu bat idazten du // Windowserako zein Linuxerako balio du // Windowserako zein Linuxerako balio du //
////////// Iratzargailu_bateragarria.c //////////////////////////////////////////////////////////////////

#include <stdio.h>
#include <stdlib.h>
#ifndef WIN32
#include <windows.h>
#else
#include <unistd.h>
#define Sleep(x) usleep((x)*1000)
#endif

unsigned itxoin_denbora (int s) {

    if (s <= 0)
        return 0;
    else {
        Sleep(s*1000);
        return(s);
    }
}

int main (int argc, const char * argv[]) {
    int i;

    if (argc < 3) {
        printf("Erabilera: %s denbora mezua\n", argv[0]);
        exit(1);
    }
    itxoin_denbora(atoi(argv[1]));
    for (i=2; i<argc; i++)
        printf("%s\n", argv[i]);
}
```

```
//////////  
// bikoizgailua.c //  
// tee komandoak egiten duen gauza bera egiten du, //  
// hau da, sarrera estandarrean idatzitakoia irteera //  
// estandarrean eta fitxategian idazten da. //  
//////////  
  
#include <unistd.h>  
#include <stdlib.h>  
#include <fcntl.h>  
  
#define BUFFER_TAMAINA 512  
  
main(int argc, const char * argv[]){  
    int n;  
    char buf[BUFFER_TAMAINA];  
    int fitx;  
  
    if(argc>2){  
        write(2,"Errorea: argumentu gehiegi\n",27);  
        exit(0);  
    }  
  
    if(argc==1){  
        while((n= read(0, buf, BUFFER_TAMAINA)) > 0)  
            write(1, buf, n);  
    }  
  
    if(argc==2){  
        fitx=open(argv[1],O_CREAT|O_RDWR|O_APPEND,00777);  
        while((n= read(0, buf, BUFFER_TAMAINA)) > 0){  
            write(fitx,buf,n);  
            write(1,buf,n);  
        }  
    }  
}
```