PRAKTIKA: C lengoaia erabiliz aplikazioak sortu (I)

Helburuak:

-Cko komandoak erabili

-Cko liburutegiak eta erazagupen edo prototipo fitxategiak

-Konpiladorearen urratsak eta erabilera
-Fitxategien kudeaketa c-ko liburutegiak erabiliz

-Liburutegi estatikoak eta dinamikoak

-Proiektuen kudeaketa: make erabiliz

Programatzeko instalatu beharreko paketeak instalatzen daki:

Instalatu hurrengo aplikazioak: liburutegi, komando eta dokumantazioa

C programatzeko , eta Cko liburutegiak erabiltzeko, hurrengo aplikazioak instalatu:
sudo apt-get update
sudo apt-get install build-essential
//programatzeko beharrezkoak diren paketeak instatzen ditu
Gainera laguntza eta dokumentazio instalatzeko:
sudo apt-get install debian-keyring
//GnuPG keys of Debian Developers
sudo apt-get install gcc-4.6-doc
//Documentation for the GNU compilers (gcc, gobjc, g++)
sudo apt-get install glibc-doc
//Embedded GNU C Library: Documentation
sudo apt-get install manpages-dev
//Manual pages about using GNU/Linux for development
POSIX komandoen laguntza erabiltzeko:
sudo apt-get install manpages-posix //Manual pages about using POSIX system
POSIX(**POSIX** es el acrónimo de **P**ortable **O**perating **S**ystem **I**nterface, y **X** viene de UNIX como seña de identidad de la API.), norma que define una interfaz estándar del sistema operativo y el entorno, incluyendo un intérprete de comandos (o "shell"), y programas de utilidades comunes para apoyar la portabilidad de las aplicaciones a nivel de código fuente. El nombre POSIX surgió de la recomendación de Richard Stallman, que por aquel entonces en la década de 1980 formaba parte del comité de IEEE.

Programatzeko laguntza tresna erabiltzen daki:

man erabili:

Gai honetan “*bash*”eko komandoak, “*c*”-ko *api*-ak eta “*linux*”eko *api*-ak erabiliko ditugu. Horretarako man komandoak ematen digun laguntza ondo etortzen da. “man” en laguntza sekzioetan banatzen da:

Sekzio Zbk- Arloa

1-.User commands edo “*bash*”eko komandoak

2-.System calls edo “SE *linux*”eko API-ak

3-.Subroutines edo “C”-ko API-ak

4-.Devices

5-.File Formats

6-.Games

7-.Miscellanous

8-.System Administration

9-.New

Erabilera: `man [opciones] [sección] <nombre>`

Adibideak:

```
$ man 1 write
$ man 2 write
$ man chmod
$ man 5 shadow
$ man -a passwd # sekzio guztietako laguntza agertzen da
```

Programatzeko tresnak erabiltzen daki:

Cko editore bat erabilitzeko:

Gedit

Ir a Ver->Modo resaltado->Fuentes->C

Fitxategi buruak eta liburutegiak kudeatzen daki?

1.-Zein da stdio.h fitxategiaren edukia? Zein katalogoan aurkitzen da?

stdio.h (standard input-output header) goiburuen fitxategia da. Hemen aurki ditzazkegu makroen definizioak, konstanteak edota eragiketak egiteko funtzioen deklarazioa.

stdio.h *//usr/include -n dago.*

2.-Non dago “printf”ren erazagupena?

stdio.h liburutegian

3.-Non daude Cko liburutegiak, bai estatikoak “libc.a”, bai dinamikoak “libc.so”?

Liburutegiak /lib eta /usr/lib –en daude

4Gehiago jakiteko irakurri Cko liburutegiari buruzko dokumentazioa:

http://www.gnu.org/software/libc/manual/html_mono/libc.html

Konpiladorearekin etapa desberdinetan sortzen diren fitxategiak kudeatzen daki:

5.-Sortu “pi.c” iturri fitxategia, hurrengo kodea sartuz:

```
#include <stdio.h>
#define PI 3.1415
int main ()
{
char c;
printf("¿Quieres conocer al número PI? (S/N)");
c=getchar();
if (c=='S' || c=='s') printf("%f",PI);
else
printf("Agur");
return 0;
}
```

Aurre-konpilatu eta aztertu irteera fitxategia **gedit pi.c**
gcc -o pi.i -E pi.c

a) Mihiztatzaile kodea konpilatu ostean, pi.s izenarekin.
gcc -o pi.s -S pi.i

b) Objetu kodea pi.o mihiztatu ostekoa, pi.o izenarekin.
gcc -o pi.o -c pi.c

c) Exekutagarria kodea estekaketaren ostean, “pi” izenarekin.
gcc -o pi pi.c
./pi

Fitxategi baten edukina irakurtzen daki:

6.-Erabiltzaileak fitxategi baten izena emanda (path-arekin) zenbat byte(karaktere) dituen pantailaratuko du.

Adibidez:
$ **gcc -o longFich longFich.c**
$ **./longFich**
¿De qué fichero deseas conocer el tamaño?longFich.c
El tamaño del fichero es de 483
$ **ls -l longFich.c**
{-rw-r--r-- 1 kepa kepa **483** 2013-03-01 12:36 longFich.c

```
#include <stdio.h>

int main(){

char fitx[255];
int kont=0;
FILE *sarrera;
printf("¿Ze fitxategiaren tamaina jakin nahi duzu?");
scanf("%s",fitx);
sarrera=fopen(fitx,"r");
if (fitx==NULL) {
        printf("Fitxategia ez da existitzen");
}
else{
char c = fgetc(sarrera);
while (c!=EOF) {
        kont=kont+1;
        c=fgetc(sarrera);
}
}
printf("%s fitxategiaren luzera %d dela",fitx,kont);
return 0;
}
```

longFich.c fitxategiaren luzera 480 da
ls -l longFich.c
-rw-r--r-- 1 endika endika 480 mar 21 11:00 longFich.c

Fitxategi baten edukina idazten daki:

7.-Erabiltzaileak fitxategi baten izena emanda (path-arekin) alfabetoa idatzi bai letra larritan eta xeheetan. Zerrenda bakoitza lerro baten idatzi.

Adibidez:

$ **gcc -o abece abece.c**
$ **./abece**
Introduce el nombre del fichero a crear: prueba.txt
$ **less prueba.txt**
abcdefghijklmnopqrstuvwxyz
ABCDEFGHIJKLMNOPQRSTUVWXYZ

```
#include <stdio.h>

int main(){

char fitx[255];
FILE * irteera;
char c;
printf("Idatzi sortuko den fitxategiaren izena: ");
scanf("%s",fitx);
irteera=fopen(fitx,"w");  //w-k sortuko du ez bada existitzen
c='a';
while (c<='z'){
        fputc(c,irteera);
        c++;
}
fputc('\n',irteera);
c='A';
while (c<='Z'){
        fputc(c,irteera);
        c++;
}
return 0;
}
```

```
gcc -o abece abece.c
./abece
Idatzi sortuko den fitxategiaren izena: prueba.txt
less prueba.txt

abcdefghijklmnopqrstuvwxyz
ABCDEFGHIJKLMNOPQRSTUVWXYZ
```

Exekutagarriak berrerabiltzen daki:

Bash-eko script bat egin, lehenengo exekutagarria ondo joanez gero, bigarren exekutagarri bat egikaritu dezala. Horretarako, aurreko programa biak erabili. “abece” exekutagarria ondo joanez gero longFich exekutagarria egikaritu bestela errore mezu bat atara.

gedit bashscript.sh

./abece

```
if [ $? -eq 0 ] ; then   //emaitza 0 balio badu orduan exekutatu hurrengo programa
   ./longFich
   echo -e "\nwell done"
   exit 0
else
   echo -e "\nfailed"
   exit 1
fi
```

chmod u+x ./bashscript.sh

./bashscript.sh

Idatzi sortuko den fitxategiaren izena: exec7.txt

¿Ze fitxategiaren tamaina jakin nahi duzu?longFich.c

longFich.c fitxategiaren luzera 480 da

well done

Fitxategiekin lan egiten daki, liburutegi propioak sortuz:

8.-Hemen dagoan programa nagusia osatu gabe dago: akopuru.c

```
#include<stdio.h>
#include “orokorra.h”
int main()
{
char nombre[80],c;
FILE *fp;
int contador=0;
printf ("¿De qué fichero deseas conocer el número de aes?");
scanf("%s",nombre);
```

```
fp=fopen(nombre,"r");
if (fp==NULL)
        {
              printf("No se puede abrir el fichero %s \n",nombre);
            return -1;
         }
else
        {       contador=ffitxkarakkop('a',fp);
                fclose(fp);
                printf("El número de a-s es de %d",contador);
        }
return 0;
}
```

a)Sortu “ffitxkarakkop” funtzioa. Funtzio honek, fitxategiaren deskriptorea eta bilatu behar duen karakterea pasa ostean, zenbat aldiz karakterea fitxategian agertzen den itzultzen du. Sartu funtzio hau “orokorra” deituko den modulu batean. Eta modulu hau “liborokorra” deituko den liburutegi dinamiko batean.

```
gedit orokorra.c
#include<stdio.h>
int ffitxkarakkop(char c, FILE *fp){
char car;
int kont=0;
car=fgetc(fp);
while (car!=EOF){
            if (car==c){
                    kont++; }
            car=fgetc(fp);  }
return kont;
}
gedit orokorra.h
#ifndef _OROKORRA_H
#define _OROKORRA_H
int ffitxkarakkop(char c, FILE *fp);
#endif
```

gcc -I. -c orokorra.c -o orokorra.o -fPIC (orokorra.c-ko objektu modulua sortu)

ld -o liborokorra.so orokorra.o -shared

b) Sortu programa printzipala liburutegi dinamokoak erabiliz. Bidezbatez osotu akopuru programa printzipala, dena ondo joan dadin.

gcc -L. -I. -o akopuru akopuru.c -Bdynamic liborokorra.so

LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/home/endika

export LD_LIBRARY_PATH

./akopuru

¿De qué fichero deseas conocer el número de aes?akopuru.c

El numero de a-s es de 14