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 fitxategiak hainbat konstante eta input eta output funtzioak ditu definituta. /usr/include/ -n dago.
2.-Non dago “printf”ren ?
Stdio.h -n dago

3.-Non daude Cko liburutegiak, bai estatikoak “libc.a”, bai dinamikoak “libc.so”?
Libc.a /usr/lib/x86_64-linux-gnu/ -n dago eta libc.so /usr/lib/x86_64-linux-gnu/ -n.
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

a) Mihiztatzaile kodea konpilatu ostean, pi.s izenarekin.
gcc pi.c -s

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

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

Fitxategi baten edukina irakurtzen daki:

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

```
#include <stdio.h>

int main(char argc, char *argv[]){
char c;
FILE* fitxategia;
int emaitza = 0;

fitxategia = fopen(argv[1],"r");
c = fgetc(fitxategia);



while(c!=EOF){
        c=fgetc(fitxategia);
        emaitza++;
}

fclose(fitxategia);
printf("\nEmaitza %d da\n", emaitza);

}
```

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

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.

```
#include <stdio.h>

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

```
    char c;
    FILE* fitxategia;

    fitxategia = fopen(argv[1],"w+");
    for (c = 'a'; c <= 'z'; ++c){
            fputc(c, fitxategia);
    }
    for (c = 'A'; c <= 'Z'; ++c){ //badakit bukle bakar batean egin daitekeela baina ez dut lortu
            fputc(c, fitxategia);
    }

    fclose(fitxategia);


    }
```

Adibidez:
$ **gcc -o abece abece.c**
$ **./abece**
Introduce el nombre del fichero a crear: 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

./**longFich**

echo -e "\nwell done"

exit 0

else

echo -e "\nfailed"

exit 1

fi

chmod u+x ./bashscript.sh

./bashscript.sh

Nire bashscript -a:

./sortu.out fitx.txt

```
if [ $? -eq 0 ] ; then
   ./longFich.out sortu.c
   echo -e "\nwell done"
   exit 0
else
   echo -e "\n failed"
   exit 1
fi
```

Fitxategiekin lan egiten daki, liburutegi propioak sortuz:

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

```
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.

Nire programa:

```
#include <stdio.h>



int ffitxkarakkop(char caractere, FILE *fitx){
int emaitza=0;
char c = fgetc(fitx);
while(c!=EOF){
        if(c==caractere){
                emaitza++;
        }
        c = fgetc(fitx);
}
return emaitza;

}

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\n",contador);
        }
    return 0;
    }
```

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

//hacer la funtzio en un archivo aparte e importala como librería

Lehenengo hiru fitxategi sortu ditugu

nagusia.c

```
#include <stdio.h>
#include "charKop.h"


int main()      {
   int contador=0;
   contador=zenbat('a');
   printf("El número de a-s es de %d\n",contador);
   return 0;
}
```

charKop.c

```
#include <stdio.h>
#include "charKop.h"


int zenbat(char caractere){
                FILE *fitx;
```

```
		char nombre[80];
		char c;
		int emaitza;

		printf ("De que fichero deseas conocer el numero de aes?");
		scanf("%s",nombre);
		fitx=fopen(nombre, "r");
		emaitza=0;
		c = fgetc(fitx);
		while(c!=EOF){
			if(c==caractere){
				emaitza++;
			}
			c = fgetc(fitx);
		}
		fclose(fitx);
		return emaitza;
}
```

charKop.h

```
#ifndef __NIRELIB__
#define __NIRELIB__

int zenbat(char caractere);

#endif
```

Eta hurrengo komandoak exekutatu:

gcc -c charKop.c -o charKop.o Liburutegia konpilatzeko

ar rv libcharKop.a charKop.o

ranlib libcharKop.a Liburutegia erabili ahal izateko

gcc -I. -L. nagusia.c -lcharKop -o proba -static “Nagusia” fitxategia konpilatzeko charKop liburutegia erabiliz (static moduan)

Eta ondoren “proba” fitxategia exekutatu

```
ander@anderSanju:~/Escritorio/c$ ./proba
De que fichero deseas conocer el numero de aes?archivo.txt
El número de a-s es de 9
```