

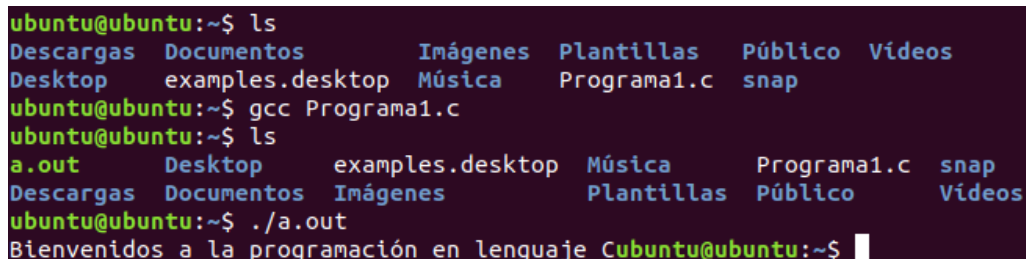
Ejercicio 1. Escriba el programa del ejemplo (líneas 1-15) usando el editor vi (o cualquier otro). Dele el nombre de “programa1.c”.

Primero escribí el código en un editor de texto y al ponerle el nombre con la terminación “.c” se colorearon las letras de distintas maneras para identificar cada parte del código. La primera línea del código es un comentario por lo cual, a la hora de ejecutar el programa, este lo ignora. La siguiente línea con “*include*” representa una directiva o una biblioteca de donde se obtienen las definiciones de algunas funciones extra para poderlas usar. La siguiente línea es la función “*main*” y dentro de su cuerpo contiene la instrucción “*printf()*” con la que se puede imprimir en pantalla lo que se escriba dentro de los paréntesis y entre comillas. Y la instrucción “*return*” que devuelve un cero (en este caso) que indicará que el programa se ejecutó y terminó exitosamente.

```
/* Este programa es con fin didáctico, solo escribe un simple mensaje */  
#include<stdio.h>  
  
int main(void)  
{  
  
    printf("Bienvenidos a la programación en lenguaje C");  
  
    return(0);  
}
```

Ejercicio 2. Compile el programa ejemplo y ejecutelo como el ejemplo de la línea anterior.

Mediante el comando “*gcc*” creé el archivo ejecutable “*a.out*” y después ejecuté ese programa en la terminal, lo que ocurrió fue que la instrucción “*printf*” se ejecutó con éxito e imprimió en pantalla “*Bienvenidos a la programación del lenguaje C*” y se terminó de ejecutar el programa. Tal cual se indicó en el código.



```
ubuntu@ubuntu:~$ ls  
Descargas  Documentos  Imágenes  Plantillas  Público  Vídeos  
Desktop    examples.desktop  Música    Programa1.c  snap  
ubuntu@ubuntu:~$ gcc Programa1.c  
ubuntu@ubuntu:~$ ls  
a.out      Desktop    examples.desktop  Música    Programa1.c  snap  
Descargas  Documentos  Imágenes          Plantillas  Público      Vídeos  
ubuntu@ubuntu:~$ ./a.out  
Bienvenidos a la programación en lenguaje Cubuntu@ubuntu:~$
```

Ejercicio 3. Repita la línea anterior. Diga lo que pasa ¿Cómo se usa el programa ejecutable?

Utilizando el comando “*gcc*” se puede hacer un ejecutable de nuestro programa en C (generalmente es un código escrito en C en un editor de textos con terminación .c). El comando “*gcc*” nos brinda un programa ejecutable para poder ejecutarlo (valga la redundancia). En nuestro caso, el programa se ejecuta llamando al ejecutable directamente a la terminal a través de la ruta relativa de

ejecutable. Sin embargo, en el momento de crear el ejecutable podemos cambiarle el nombre que viene por default (*a.out*) a través de la opción “-o” junto al comando “*gcc*”. Esta opción hace que podamos ponerle nombre, el cual se colocará inmediatamente después de la opción y seguido de eso se coloca el nombre el archivo con terminación “.c”.

El profesor indica en la practica que, por lo general, al ejecutable se le pone el nombre del archivo donde está el código pero sin la terminación “.c”

```
ubuntu@ubuntu:~$ ls
Descargas  Documentos  Imágenes  Plantillas  Público
Desktop    examples.desktop  Música    Programa1.c  Vídeos
ubuntu@ubuntu:~$ gcc Programa1.c
ubuntu@ubuntu:~$ ls
a.out      Desktop    examples.desktop  Música    Programa1.c  Vídeos
Descargas  Documentos  Imágenes          Plantillas  Público
ubuntu@ubuntu:~$ gcc -o Programa1 Programa1.c
ubuntu@ubuntu:~$ ls
a.out      Desktop    examples.desktop  Música    Programa1  Público
Descargas  Documentos  Imágenes          Plantillas  Programa1.c  Vídeos
ubuntu@ubuntu:~$
```

Y bueno, como se mencionó antes, para ejecutar el ejecutable solo se coloca la ruta relativa del ejecutable en la terminal.

```
ubuntu@ubuntu:~$ ls
a.out      Desktop    examples.desktop  Música    Programa1  Público
Descargas  Documentos  Imágenes          Plantillas  Programa1.c  Vídeos
ubuntu@ubuntu:~$ ./Programa1
Bienvenidos a la programación en lenguaje C
ubuntu@ubuntu:~$
```

Instrucción printf

Ejercicio 4. Siga la explicación anterior y después de hacer los cambios correspondientes ejecute el nuevo programa. Anote lo visto con los cambios. ¿Qué diferencia notó?

Modifiqué el archivo para colocar “\n” dos veces al final de lo que escribí dentro de “*printf*” quedando así:

```
/* Este programa es con fin didáctico, solo escribe un simple mensaje */
#include<stdio.h>

int main(void)
{
    printf("Bienvenidos a la programación en lenguaje C\n\n");

    return(0);
}
```

Lo que significa “\n” es que se da un salto de línea al momento de ejecutar el programa. En este caso, se darán dos saltos de línea después de que se imprima en pantalla “*Bienvenidos a la programación en C*”.

Nuevamente, se guarda el archivo y se crea el ejecutable (esta vez bajo el nombre “Programa2”) y después se ejecuta el programa en la terminal y estos son los resultados:

```
ubuntu@ubuntu:~$ ./Programa1
Bienvenidos a la programación en lenguaje Cubuntu@ubuntu:~$
```

```
ubuntu@ubuntu:~$ ./Programa2
Bienvenidos a la programación en lenguaje C
ubuntu@ubuntu:~$
```

La diferencia entre los dos programas es justamente los saltos de línea, en el Programa1 no se realizan saltos de línea y, por lo tanto, la siguiente instrucción que espera que ingreses la terminal se queda sobre la misma línea. Mientras que en el Programa2 la línea que espera que se le ingrese una instrucción después de haber ejecutado y terminado la instrucción anterior (en este caso después de que se termine de ejecutar el programa que hemos creado).

Esto es medio confuso pero se observa mejor si ponemos más secuencias de escape ("*printf*") y observamos los resultados.

Modificando un poco el código del programa se puede ver que si no se colocan los saltos de línea, entonces entre las propias instrucciones del código no hay salto de línea al momento de ejecutarlo en la terminal, esto se observa aquí, donde Se imprimen dos oraciones pero sin salto de línea, es decir, las dos oraciones se imprimen sobre la misma línea.

Código del ejemplo1:

```
/* Este programa es con fin didáctico, solo escribe un simple mensaje */
#include<stdio.h>

int main(void)
{
    printf("Hola, mi nombre es Cristian Pichardo.");
    printf("Hola, mi nombre es Juan Fernandez.");

    return(0);
}
```

Programa ejecutado:

```
ubuntu@ubuntu:~$ ./ejemplo1
Hola, mi nombre es Cristian Pichardo.Hola, mi nombre es Juan Fernandez.ubuntu@ubuntu:~$
```

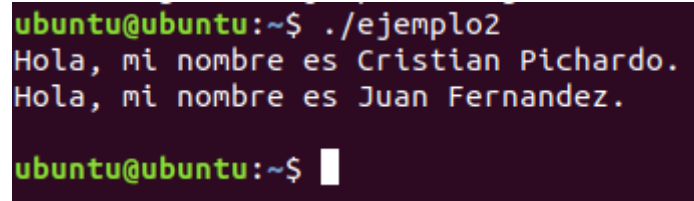
Mientras que si se utiliza correctamente "*\n*" entonces habrá saltos de línea y se puede tener mas control sobre el código.

En este caso, el primer "*printf*" solo realiza un salto de línea mientras que el segundo "*printf*" manda dos saltos de línea.

Código del ejemplo2:

```
/* Este programa es con fin didáctico, solo escribe un simple mensaje */  
  
#include<stdio.h>  
  
int main(void)  
{  
  
    printf("Hola, mi nombre es Cristian Pichardo. \n");  
    printf("Hola, mi nombre es Juan Fernandez.\n\n");  
  
    return(0);  
}
```

Programa ejecutado:



```
ubuntu@ubuntu:~$ ./ejemplo2  
Hola, mi nombre es Cristian Pichardo.  
Hola, mi nombre es Juan Fernandez.  
  
ubuntu@ubuntu:~$
```

Ejercicio 5. Algunas de las secuencias de escape usadas en clases anteriores se muestran abajo. Úselas:

Bueno, ya vimos que “\n” nos da un salto de línea, “\”” nos coloca las comillas, “\”” nos coloca las comillas simples, “\\” nos coloca la barra inclinada, “\v” nos da una tabulación vertical, “\r” nos da un regreso de carro al principio de la línea, “\b” nos da un retroceso, “\f” nos da un salto de página y “\t” nos brinda una tabulación horizontal. Veamos nuestro ejemplo:

```
/*uso de las sevuencias de escape en C */  
  
#include <stdio.h>  
  
int main(void) {  
    printf("Según muchos argentinos, Maradona era el \"Dios\" del futbol. \n");  
    printf("Dirección: C\\ Mayor 25 \n");  
    printf("Ha salido el número \'25\' \n");  
    printf("Retroceso \b \n \v");  
    printf("\a Holaaaaa. \n \f");  
    printf("\r Amigos míos.");  
    printf("\t Esto ha sio todo. \n");  
    return(0);  
}
```

Este código, al momento de ejecutarse, primero debería imprimir en pantalla “Según muchos argentinos, Maradona era el “Dios” del fútbol” seguido de un salto de línea, ya que \” nos coloca las comillas y \n nos da un salto de línea. Después debería imprimir “Dirección: C/Mayor 25” seguido de un salto de línea, esto por \n (que para hacerlo mas amo ya no se explicara) y por \\ que coloca la barra inclinada, después se debería imprimir “Ha salido el número ‘25’” seguido de un salto de línea, esto porque \' imprime las comillas sencillas. Seguidamente se imprimiría “Retroceso” seguido de un retroceso (por \b), un salto de línea y una tabulación vertical (por \v). Después de la tabulación vertical se imprimiría una campana (o por como la interpreto, una sangría) y después “Holaaaaa” seguido de un salto de línea y de un salto de página (por \f), luego de este salto se imprimiría un espacio en blanco (debido a \r) y seguidamente “Amigos míos.” luego una tabulación debido a \t y “Esto ha sido todo” junto con un salto de línea. Como vemos en la siguiente imagen, esto fue lo que ocurrió:

```

ubuntu@ubuntu:~$ gcc -o A A.c
ubuntu@ubuntu:~$ ./A
Según muchos argentinos, Maradona era el "Dios" del futbol.
Dirección: C\ Mayor 25
Ha salido el número '25'
Retroceso

Holaaaaaa.

Amigos míos. Esto ha sido todo.
ubuntu@ubuntu:~$

```

Ejercicio 6. En cada uno de los programas explique que hacen:

El primer programa es el que hicimos al principio, en general, solo imprime todo lo que está dentro de la función “printf()”.

```

/* prog01.c */
/* Programa No.1 */
/* Este programa es el primero y se llama prog01.c */
/* Muestra el uso de la función printf() */

#include <stdio.h>

int main(void)
{
    printf("¡Bienvenidos al lenguaje programación en C! \n");
    return(0);
}

```

```

ubuntu@ubuntu:~$ gcc -o A prog01.c
lsubuntu@ubuntu:~$ ls
A          Documentos      Música      Programa1.c  Vídeos
Descargas  examples.desktop Plantillas  Público
Desktop    Imágenes         prog01.c   snap
ubuntu@ubuntu:~$ ./A
¡Bienvenidos al lenguaje programación en C!
ubuntu@ubuntu:~$

```

El segundo programa se llama prog02.c aunque el profesor se confundió y en la segunda línea colocó que este programa era prog03.c. Este programa solamente agrega líneas de salto (3 en lugar de una). Un ejercicio que también fue explicado más arriba.

```

/* prog02.c */
/* Programa No.2 */
/* Este programa es el primero y se llama prog03.c */
/* Observe la diferencia */

#include <stdio.h>

int main(void)
{
    printf("¡Bienvenidos al lenguaje programación en C! \n\n\n");
    return(0);
}

```

```
ubuntu@ubuntu:~$ gcc -o B prog02.c
ubuntu@ubuntu:~$ ./B
¡Bienvenidos al lenguaje programación en C!

ubuntu@ubuntu:~$
```

El tercer programa (que el profesor llama como cuarto programa), esta vez en pantalla solo se imprime “¡Hola a todos!” seguido de un salto de página (por \f).

```
/* prog04.c */
/* Programa No.4 */
/* Este programa es nl primero y se llama prog04.c */

#include <stdio.h>

int main(void)
{
    printf("¡Hola a todos! \f");
    return(0);
}
```

```
ubuntu@ubuntu:~$ gcc -o C prog04.c
ubuntu@ubuntu:~$ ./C
¡Hola a todos!

ubuntu@ubuntu:~$
```

Y el cuarto programa solo imprime 3 oraciones diferentes utilizando tres funciones, sin dejar saltos de línea entre cada oración lo que a la hora de ejecutar el programa se ve como una oración continua, con excepción del final que tiene dos saltos de línea.

```
/* prog04.c */
/* Programa No.4 */
/* Este programa es nl primero y se llama prog04.c */

#include <stdio.h>

int main(void)
{
    printf("¡Hola a todos!");
    printf(" Hoy les toca clase y ");
    printf("práctica \n\n");
    return(0);
}
```

```
ubuntu@ubuntu:~$ gcc -o D prog04.c
ubuntu@ubuntu:~$ ./D
¡Hola a todos! Hoy les toca clase y práctica

ubuntu@ubuntu:~$
```