PROGRAMACIÓN EN C

Ejercicio 1

ENUNCIADO

Escribe un programa en C que nos avise si una determinada cadena de caracteres está incluida en una o varias líneas que vayamos escribiendo por teclado. La cadena que buscar se debe pasar por parámetro al ejecutar el programa de manera que pondremos en línea de comandos:

> *NombreEjecutable CadenaQueBuscar*

SOLUCIÓN PROPUESTA

Vamos a desarrollar la solución en el entorno donde normalmente se nos solicita en la parte práctica en las oposiciones que es en un entorno Linux.

Siempre suponemos que los equipos donde se realizan las pruebas están preparados para realizarlas sin tener que instalar nada pero para tener una visión global de la solución y para realizar este ejercicio desde cero, explicaré paso a paso como si no tuviéramos nada instalado y dirigido para los que no dominan este tipo de ejercicios.

Para ver si tenemos la herramienta que compila el código C en nuestro sistema, desde un terminal de Linux podemos introducir el siguiente comando:

```
root@usuario-VirtualBox:/# gcc --version
gcc (Ubuntu 9.4.0-1ubuntu1~20.04.2) 9.4.0
Copyright (C) 2019 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

Si no sale el mensaje anterior, nos indicará que debemos instalarla con la siguiente orden:

```
# apt install gcc
```

Una vez instalada, procederemos a escribir el código que realice lo que nos pide el ejercicio. Ya que el ejercicio no indica el nombre del archivo, nuestro archivo se llamará *contiene.c*. Para ello en la misma terminal podemos utilizar el editor de textos *nano* que con la opción *-l*, se nos muestra los números de línea, muy útiles para localizar cualquier error que pueda surgir dentro del código:

```
# nano -l contiene.c
```

```
GNU nano 4.8                          contiene.c                          Modificado
1  #include <stdio.h> /* gets, puts */
2  #include <string.h> /* strstr*/
3
4  #define MAX_LINE 8192
5
6  int main (int argc, char *argv[])
7  {
8          char line[MAX_LINE];
9
10         if (argc!=2) {
11                 printf("Uso: %s cadena\n",argv[0]);
12         } else {
13                 while (gets(line)!=NULL) {
14                         if (strstr(line,argv[1])!=NULL) {
15                                 printf("Cadena encontrada\n");
16                         }
17                         else
18                                 printf("Cadena NO encontrada\n");
19                 }
20         }
21
22         return 0;
23 }
```

El paso siguiente sería compilar el código con la herramienta anteriormente instalada:

```
# gcc -o contiene contiene.c
```

Utilizamos la opción *-o* para indicarle el nombre del ejecutable, en este caso *contiene.*

Tal y como indica el ejercicio el programa debe aceptar parámetros, así que una vez esté el ejecutable listo, lo probamos:

```
# ./contiene hola
```

```
Cadena NO encontrada
Aqui no está la cadena pasada como parámetro
Cadena NO encontrada
Aquí si está hola
Cadena encontrada
hola hola
Cadena encontrada
Aquí no
Cadena NO encontrada
```

Hay que pulsar CTRL+D para que el programa detecte el fin de entrada o null que hemos indicado en el programa para la finalización de la ejecución.

A continuación, procederemos a la explicación exhaustiva del código:

#include <stdio.h>

- **Significa:** Incluir la biblioteca estándar de entrada/salida.
- **¿Para qué sirve?** Nos permite usar funciones como:
  - printf() para mostrar texto en pantalla.
  - gets() para leer del teclado (entrada estándar).

#include <string.h>

- **Significa:** Incluir la biblioteca de manejo de cadenas (strings).
- **¿Para qué sirve?** Permite usar funciones como:
  - strstr() → busca una subcadena dentro de otra.

#define MAX_LINE 8192

- **Significa:** Crear una **constante llamada MAX_LINE con valor 8192**.
- **¿Para qué sirve?** Define el **máximo número de caracteres** que puede tener cada línea que leeremos.

---

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

- **Esta es la función principal del programa que debe admitir argumentos o parámetros en su llamada desde la consola.**
  > *NombreEjecutable CadenaQueBuscar*
- **argc**: número de argumentos que se han pasado al programa desde la consola .
- **argv[]**: un array con el nombre del programa y los argumentos en forma de texto.
  - argv[0]: es el nombre del programa.
  - argv[1]: será la palabra que queremos buscar.

---

char line[MAX_LINE];

- Crea un **array de caracteres** llamado line, capaz de almacenar hasta 8192 caracteres (una línea de texto).
- Se usa para guardar la línea que el usuario escribe.

---

if (argc != 2)

- Verifica si **el usuario no ha pasado una palabra como argumento** al ejecutar el programa.
- Si argc != 2, es porque falta el texto a buscar, argc debe tener exactamente el valor 2, de 2 elementos en argv[], el nombre del programa y la cadena que buscar.
- En ese caso, mostramos un mensaje indicando el uso correcto.

---

while (gets(line) != NULL)

- **gets**: lee una línea desde el teclado y la guarda en **line**.
- Se repite mientras no se llegue al **fin de entrada (EOF)**.
  - En Linux se logra pulsando Ctrl + D.
  - En Windows, con Ctrl + Z seguido de Enter.

---

if (strstr(line, argv[1]) != NULL)

- strstr(a, b): busca la subcadena b dentro de a.
- Si devuelve algo distinto de NULL, es que **se encontró** la palabra que buscábamos (argv[1]).
- Si no devuelve NULL, que significa que **no está** en la línea.

---

printf("Cadena encontrada\n"); o printf("Cadena NO encontrada\n");

- Muestra por pantalla si la palabra fue o no encontrada en la línea introducida.

---

return 0;

- Indica que el programa finalizó **correctamente**.