Seminario de lenguajes
Opción C

Práctica 2 - 2021

1. Analice en su sistema operativo cuál es la combinación de teclas que da como resultado EOF. Analice cómo cambia esta combinación en GNU/Linux y Windows.

2. Escriba un programa que verifique que la expresión **getchar() != EOF** sea 1 o 0.

3. Escriba un programa que imprima el valor de EOF. ¿Alcanza el tipo **char** para almacenar EOF? ¿Sería correcto el siguiente extracto de código?

```
char c;

c = getchar();
while (c != EOF) {
```

4. Escriba un programa que lea caracteres del teclado empleando **getchar()** y los cuente hasta encontrar **EOF**. Luego debe imprimir la cantidad de caracteres y líneas encontradas. *En GNU/Linux puede comprobar la funcionalidad con los comandos **wc -c** y **wc -l**.*

   Nota: Puede leer de teclado o invocar el programa con un archivo cómo parámetro. Por ejemplo:

```
./ejecutable.bin < archivo_param
```

   Esto hace que el contenido de **archivo_param** se vea desde el programa **ejecutable.bin** como proveniente del teclado.

   Notará que si utiliza sólo el teclado, el programa no funcionará hasta que ingrese ENTER. Esto es propio del manejo de buffers en C.

5. Escriba un programa que copie caracteres del teclado en la pantalla reemplazando cada ocurrencia de uno o más espacios por sólo uno. Emplee **getchar()**.

6. Escriba un programa que copie caracteres del teclado en la pantalla remplazando cada ocurrencia de barra invertida por \\, cada tabulación por \t, cada enter por \n. Emplee **getchar()**.

```
# Por ejemplo. Si ingresa:
hola___mundo
chau mundo\
# Debe imprimir:
hola\tmundo\nchau mundo\\
```

7. Pruebe con números enteros la diferencia de usar **printf** de las siguientes formas:
   (a) %d
   (b) %10d
   (c) %-10d

8. Pruebe con números reales la diferencia de usar **printf** de las siguientes formas:
   (a) %f
   (b) %10f
   (c) %-10f

9. Analice qué es lo que imprime el siguiente fragmento de código:

```
char *str = "Hello world";
printf("%.*s\n", 5, str);
```

10. Modifique del siguiente código, el string de formato a la función **printf(DEF)** a fin de lograr una impresión similar a la mostrada.

```
#include <stdio.h>

int main()
{
   const char* dias[7] = {
      "Lunes", "Martes", "Miercoles", "Jueves",
      "Viernes", "Sabado", "Domingo"};

   int i;
   for (i = 0; i < 7; i++)
      printf(DEF, 3, dias[i]);

   for (i = 1; i <= 31; i++) {
      if (!((i - 1) % 7))
        printf("\n");

      printf(DEF, i);
   }

   printf("\n");

   return 0;
}
```

Resultado esperado

```
Lun Mar Mie Jue Vie Sab Dom
  1   2   3   4   5   6   7
  8   9  10  11  12  13  14
 15  16  17  18  19  20  21
 22  23  24  25  26  27  28
 29  30  31
```

11. Analice la función **scanf**:
   (a) ¿Cuál es su valor de retorno?
   (b) ¿Qué hacen las siguientes invocaciones a **scanf**?

```
int x, y;
scanf("%d", &x);
scanf("%d %d",&x,&y);
scanf("%d%d",&x,&y);
```

(c) Explique la importancia del uso de & en el punto anterior.

(d) Dado el siguiente programa que lee fechas en formato dd/mm/yyyy, analice qué sucede si envía como entrada una letra en vez de un número. Analice una solución a partir de la funcionalidad de **scanf**.

```
#include <stdio.h>

int main()
{
   int res, dia, mes, anio;
   do {
      res = scanf("%2d/%2d/%4d", &dia, &mes, &anio);
      printf("scanf retorno %d\n", res);

      if (res != EOF) {
         if (res != 3) {
            printf("ERROR: El formato debe ser dd/mm/yyyy\n");
         } else {
            printf("Fecha: %d/%d/%d\n", dia, mes, anio);
         }
      }
   } while (res != EOF);

   return 0;
}
```

(e) Una vez solucionado el problema del código anterior, pruébelo con las siguientes entradas.

```
1234/11/1234
12/11/1234
111/222/1234
1/11/123411/11/1234
```

12. Dado el siguiente código:

```
int main()
{
   return 0;
}
```

(a) Utilice **gcc -E** para verificar su salida

(b) Modifique el programa agregando el siguiente fragmento de código:

```
#ifndef RETORNO
#define RETORNO 10
#endif
int main()
{
   return RETORNO;
}
```

(c) Verifique con **gcc -E**. Luego verifique con **gcc -E -DRETORNO=3**

(d) Modifique nuevamente el código agregando lo siguiente:

```c
#include <stdio.h>

int main()
{
   return 0;
}
```

(e) Verifique con **gcc -E**

13. Defina dos macros: **min**, para calcular el mínimo entre dos números, y **max**, para calcular el máximo.
    (a) Analice los efectos laterales de invocar la macro **min(i++, j++)** o **max(i++, j++)**
14. Dada la siguiente macro:

```c
#define cuadrado(x) x*x
```

    (a) Analice los efectos laterales de invocar la macro **cuadrado(x + 1)**
15. Indique qué es lo que hacen las siguientes macros:

```c
#define macro1(expr) printf(#expr "= %g\n", expr);
#define macro2(uno, dos) uno ## dos
```

16. Indique qué es lo que hace la siguiente macro:

```c
#define mi_macro(t, a, b) {t _z##a##_##b = a; a = b; b = _z##a##_ ##b ;}
```

Ejercicio adicional

Los siguientes programas tienen distintos tipos de errores, algunos son difíciles de detectar y es probable que necesite ayuda para encontrarlos.

Páselos a máquina, pruébelos para encontrar los errores y discútalos con los ayudantes:

gets.c

```c
#include <stdio.h>

int main(){
      int x = 1300;
      char buffer[4];
      printf("x = %d\n", x);
      printf("Ingresa por teclado: \"hola\"\n");
      gets(buffer);
      printf("x = %d\n", x);
      return 0;
}
```

scanf.c

```c
#include <stdio.h>
#include <stdlib.h>
/*
 Probar con distintas entradas, por ejemplo:
  queso 1
  2\n5
  5 (EOF)
  64\n(EOF)
 En Linux EOF es Ctrl+D y en Windows es Ctrl+Z
*/

void error_de_lectura(int codigo){
	int caracter;
	if (codigo == EOF){
		puts("Se alcanzo fin de archivo");
		exit(1);
	}
	else{
		puts("Error de conversion, limpiando buffer...");
		do {
			caracter = getchar();
		} while (caracter != EOF && caracter != '\n');
	}
}

int main(int argc, char *argv[]){
	int n1, n2;
	int leidos;
	for (int i = 0; i < 10; i++){
		printf("Ingrese 2 numeros: ");
		if ((leidos = scanf("%d %d", &n1, &n2)) != 2){
			error_de_lectura(leidos);
			continue;
		}
		printf("%d + %d = %d\n", n1, n2, n1 + n2);
	}
	return 0;
}
```

scanf2.c

```c
#include <stdio.h>

/****************** Casos de prueba:
Ingrese al menos el siguiente conjunto de datos para probar el programa.
-> Caso 1:
marcos 54
francisco 23
analia 21
mariana 10
federico 20

-> Caso 2:
marcos pardo 24
francisco 10
analia 21.2
mariana 23
federico 20
```

```
Lea: http://c-faq.com/stdio/scanfprobs.html y las 3 preguntas relacionadas
    enlazadas al principio del documento.
**********************************/

int main(int argc, char **argv){
      char nombre[20];
      int edad;
      int error;

      // Lee e imprime 5 nombres seguidos de su edad
      for (int i = 0; i < 5; i++){
            printf("Ingrese el nombre y la edad: ");
            if ((error = scanf("%s %d", nombre, &edad)) != 2){
                  printf("------> Ocurrio un error, scanf retorno: %d\n", error);
                  // Si falla descartamos el intento
                  i--;
            }
            else{
                  printf("Ingreso el nombre: %s con edad: %d\n", nombre, edad);
            }
      }
}
```

enteros1.c

```
#include <stdio.h>

int main(int argc, char **argv){
      short x = 245;
      short y = 500;
      short z = x * y;
      printf("%hd * %hd = %hd\n", x, y ,z);
      return 0;
}
```

enteros2.c

```
#include <stdio.h>

int main(int argc, char **argv){
      unsigned i;
      for (i = 10; i >= 0; i--){
            printf("Valor de i = %u\n", i);
      }
      return 0;
}
```

enteros3.c

```
#include <stdio.h>
#include <math.h>
int main(int argc, char **argv){
      printf("2 elevado a la quinta es = %d\n", pow(2, 5));
      return 0;
}
```