

## **PRÁCTICA 2**

**PRESENTADO POR:  
DIANA MARCELA LOZADA GARCÍA**

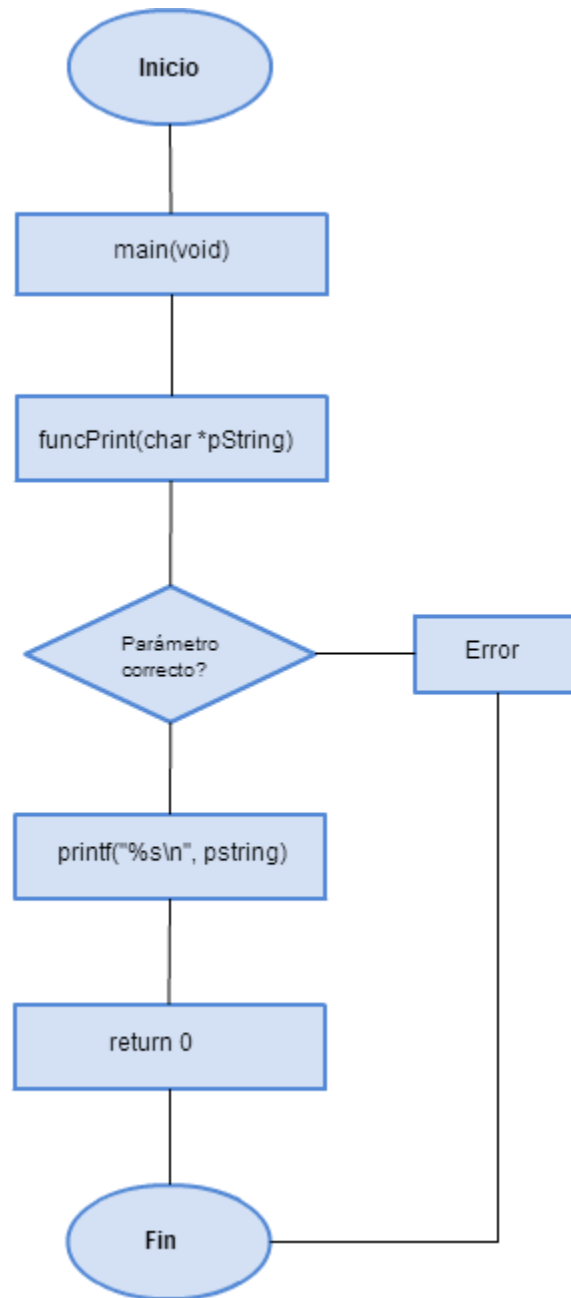
**PROFESOR:  
HENRY ARCILA**

**ASIGNATURA:  
LABORATORIO DE SISTEMAS OPERATIVOS**

**UNIVERSIDAD DE ANTIOQUIA  
DEPARTAMENTO DE INGENIERÍA DE SISTEMAS  
MEDELLÍN  
2012**

## CONTINUACIÓN PRÁCTICA 2

6.

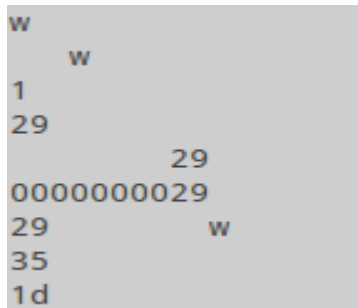


13.

- **Código:**

```
1 #include <stdio.h>
2
3 int main(void)
4 {
5     char lett = 'w';
6     int i = 1;
7     int j = 29;
8
9     printf("%c\n", lett);
10    printf("%4c\n", lett);
11    printf("%d\n", i);
12    printf("%d\n", j);
13    printf("%10d\n", j);
14    printf("%010d\n", j);
15    printf("%-10d%c\n", j, lett);
16    printf("%20\n", j);
17    printf("%2x\n", j);
18    return(0);
19 }
```

- **Resultado:**



```
w
  w
1
29
          29
0000000029
29          w
35
1d
```

- **Interpretación del resultado:**

**Línea 1: w**

Imprime un carácter simple. Seguido, imprime nueva línea.

**Línea 2: w**

Imprime el carácter w en una línea de ancho 4. El ancho se indica ubicando un número entero entre el carácter % y el carácter c. Seguido, imprime nueva línea.

**Línea 3: 1**

Imprime un entero decimal desde la variable i. Seguido, imprime nueva línea.

**Línea 4: 29**

Imprime un entero decimal desde la variable j. Seguido, imprime nueva línea.

**Línea 5: 29**

Imprime una línea de ancho 10, la cual tiene al final el número decimal 29, el cual procede de la variable j. Seguido, imprime nueva línea.

**Línea 6: 0000000029**

Imprime una línea que rellena el espacio disponible del ancho con ceros y deja alineado al lado derecho el número decimal. Seguido, imprime nueva línea.

**Línea 7: 29      w**

Imprime las variables j y lett alineadas la una a la derecha y la otra a la izquierda, separadas por 10 espacios. Seguido, imprime nueva línea.

**Línea 8: 35**

Imprime un número entero octal sin signo. Seguido, imprime nueva línea.

**Línea 9: 1d**

Imprime un número hexadecimal sin signo. Seguido, imprime nueva línea.

14.

- **Código:**

```
1 #include <stdio.h>
2
3 int main(void)
4 {
5     float x = 333.123456;
6     double y = 333.1234567890123456;
7     char lett = 'w';
8
9     printf("%f\n",x);
10    printf("%.1f\n",x);
11    printf("%20.3f\n",x);
12    printf("%-20.3f%c\n",x,lett);
13    printf("%020.3f\n",x);
14    printf("%.9f\n",y);
15    printf("%.20f\n",y);
16    printf("%20.4f\n",y);
17    return(0);
18 }
```

- **Resultado:**

```
333.123444
333.1
          333.123
333.123          w
00000000000000333.123
333.123456789
333.12345678901232304270
          333.1235
```

- **Interpretación del resultado:**

**Línea 1: 333.123444**

Imprime el número flotante simple sin ninguna especificación en especial.

**Línea 2: 333.1**

Imprime el número flotante con especificación de 1 decimal, especificado después del % como .1 y antes de f.

**Línea 3:            333.123**

Imprime el número flotante con especificación de 3 decimales y una línea de ancho de 20. Esta especificación esta entre % y f.

**Línea 4: 333.123                      w**

Imprime el número flotante con especificación de 3 decimales, e imprime un carácter w el cual está separado con una distancia de 20 con el primer dato del primer numero.

**Línea 5: 00000000000000333.123**

Imprime el número flotante con una especificación de 3 decimales y ancho de 20 llenando los espacios con ceros en la especificación 020.3

**Línea 6: 333.123456789**

Imprime el número flotante con especificación de 9 decimales, especificado después de el % como .9 y antes de f.

**Línea 7: 333.123456789**

Imprime el número flotante con especificación de 20 decimales, especificado después del % como .20 y antes de f.

**Línea 8:                      333.1235**

Imprime el numero flotante con especificación de 4 decimales y con un ancho de 20, esta especificación al igual que las anteriores esta entre % y f que indica q es flotante.

**15.**

**Código:**

```
1 #include <stdio.h>
2
3 int main(void)
4 {
5
6     char s[] = "an evil presence";
7     printf("%s\n", s);
8     printf("%7s\n", s);
9     printf("%20s\n", s);
10    printf("%-20s\n", s);
11    printf("%.5s\n", s);
12    printf("%.12s\n", s);
13    printf("%15.12s\n", s);
14    printf("%-15.12s\n", s);
15    printf("%3.12s\n", s);
16    return(0);
17 }
```

• **Resultado:**

```
an evil presence
an evil presence
    an evil presence
an evil presence
an ev
an evil pres
    an evil pres
an evil pres
an evil pres
```

- **Interpretación del resultado:**

**Línea 1: an evil presence**

Imprime la cadena de caracteres contenida en el arreglo s[].Seguido, imprime nueva línea.

**Línea 2: an evil presence**

Imprime una cadena de caracteres proveniente del arreglo s[] proporcionando un espacio de ancho 7. Dado que este valor del ancho es menor que la longitud de la cadena a imprimir, el efecto de la alineación a la derecha no es visible.

**Línea 3: an evil presence**

Imprime una cadena de caracteres proveniente del arreglo s[] proporcionando un espacio de ancho 20 y alineando estos a la derecha.

**Línea 4: an evil presence**

Imprime una cadena de caracteres proveniente del arreglo s[] proporcionando un espacio de ancho 20 y alineando estos a la izquierda.

**Línea 5: an ev**

Imprime una cadena de caracteres proveniente del arreglo s[] con una longitud de máximo 5 caracteres.

**Línea 6: an evil pres**

Imprime una cadena de caracteres proveniente del arreglo s[] con una longitud de máximo 12 caracteres.

**Línea 7: an evil pres**

Imprime una cadena de caracteres proveniente del arreglo s[] con una longitud de máximo 12 caracteres dejando un espacio de ancho 15.

**Línea 8: an evil presence**

Imprime la cadena de caracteres dejando un ancho de 15 espacios hacia la derecha, especificando que solo muestre 12 caracteres.

**Línea 8: an evil presence**

Imprime la cadena de caracteres dejando un ancho de 3 y especificando que se muestren únicamente los 12 primeros caracteres.

**16.**

El siguiente código se imprime el signo % y nueva línea:

```
1 #include <stdio.h>
2
3 int main(void)
4 {
5     printf("%%\n");
6     return(0);
7 }
```

17.

```
#include <stdio.h>

int main(void)
{
    int a;
    int b;
    int operacion;
    int resultado;
    printf ("Ingrese el Primer Operando:\n");
    scanf("%i",&a);
    printf ("Ingrese el Segundo Operando:\n");
    scanf("%i",&b);
    printf ("¿Qué operación desea realizar?: \n1:suma,\n2:resta,\n3:multiplicacion,\n4:division\n");
    scanf("%i",&operacion);
    switch(operacion){
        case 1 :
            resultado=a+b;
            printf("Resultado de la suma: %i\n",resultado);
            break;
        case 2 :
            resultado=a-b;
            printf("Resultado de la resta: %i\n",resultado);
            break;
        case 3 :
            resultado=a*b;
            printf("Resultado de la multiplicación: %i\n",resultado);
            break;
        case 4 :
            if(b!=0)
            {
                resultado=a/b;
                printf("Resultado de la división: %i\n",resultado);}
            break;
        default:
            printf("Esa operación no existe.. chau\n");
    }
    return(0);
}
```

```
Ingrese el Primer Operando:
1
Ingrese el Segundo Operando:
1
¿Qué operación desea realizar?:
1:suma,
2:resta,
3:multiplicacion,
4:division
1
Resultado de la suma: 2
```