

# EJERCITACIÓN Nº 2: PROGRAMANDO EN C

Cátedra Programación II

Octubre 2016

## 1. Switch

**EJERCICIO 1.** Escriba un programa que pida por teclado el resultado obtenido al lanzar un dado de seis caras y que muestre por pantalla el número en letras de la cara opuesta al resultado obtenido. En caso que el valor ingresado no sea válido para las caras de un dado, se debe mostrar el mensaje: "Número incorrecto".

**EJERCICIO 2.** En la siguiente tabla se muestra el número de camas de las habitaciones de una casa de campo y la planta donde está ubicada cada una de ellas:

Habitación	Camas	Planta
1. Azul	2	Primera
2. Roja	1	Primera
3. Verde	3	Segunda
4. Rosa	2	Segunda
5. Gris	1	Tercera

Se pide que escriba un programa que:

1. Muestre el listado de las habitaciones de la casa de campo.
2. Pida por teclado el número (dato entero) asociado a una habitación.
3. Muestre por pantalla la planta y el número de camas de la habitación seleccionada.

Observación: Si el número introducido por el usuario, no está asociado a ninguna habitación, se mostrará el mensaje: "Número no asociado a habitación."

## 2. Estructura For

**EJERCICIO 3.** Calcule mediante bucles for las siguientes sumatorias.


$$\sum_{n=1}^{100} \frac{1}{n}$$

$$\sum_{k=1}^{30} \frac{1}{k^2}$$

$$\sum_{j=1}^{25} \frac{1}{j^j}$$

$$\sum_{i=2}^{10} (i+1)i$$

**EJERCICIO 4.** Una terna de números naturales  $(a, b, c)$  es una *terna pitagórica* si  $a^2 + b^2 = c^2$ . Escriba un programa que imprima todas las ternas pitagóricas con  $a \leq 20$  y  $b \leq 30$ .

**EJERCICIO 5.** ¿Qué hace el siguiente programa?

```
#include <stdio.h>
int main() {
    int x, y;
    printf ("Ingrese dos enteros dentro del rango [1,20]:\n");
    scanf ("%d %d", &x, &y);
    if (x >=1 && y>=1 && x<=20 && y<=20) {
        for (int i=1; i<=y; i++) {
            for (int j=1; j<=x; j++) {
                printf ("@");
            }
            printf ("\n");
        }
    } else {
        printf("Los enteros no se encuentran en el rango pedido");
    }
    return 0;
}
```

**EJERCICIO 6.** Escriba un programa que tenga un número secreto entre 0 y 500 el cuál estará fijo (use `#define` para esto). El usuario deberá, mediante el ingreso de números, adivinar dicho valor, en cada intento el programa debe responder *el número es mayor o el número es menor*, según corresponda. El usuario dispondrá como máximo de 15 intentos.

**EJERCICIO 7.** Mediante una única instrucción `for` y un `printf` genere la siguiente salida. Emplee variables para la cadena de texto, un entero y un `char`:

b 5 T

bu 4 s

buc 3 R

buc1 2 q

bucle 1 P

**EJERCICIO 8.** Leer un valor positivo, y hacer la siguiente secuencia: si el número es par, dividirlo entre 2; si es impar, multiplicarlo por 3 y sumarle 1. Repetir lo anterior hasta que el valor sea 1, imprimiendo cada valor, también se deberá imprimir cuantas operaciones de estas son hechas en cambio, si el valor ingresado es menor que 1, imprimir un mensaje que contenga la palabra **Error**. Un ejemplo de la salida del programa podría ser el siguiente:

El valor inicial es 9

El siguiente valor es 28

El siguiente valor es 14

El siguiente valor es 7

El siguiente valor es 22  
El siguiente valor es 11  
El siguiente valor es 34  
El siguiente valor es 17  
El siguiente valor es 52  
El siguiente valor es 26  
El siguiente valor es 13  
El siguiente valor es 40  
El siguiente valor es 20  
El siguiente valor es 10  
El siguiente valor es 5  
El siguiente valor es 16  
El siguiente valor es 8  
El siguiente valor es 4  
El siguiente valor es 2  
Valor final 1, número de pasos 19.

### 3. Arrays

**EJERCICIO 9.** Escriba un programa que complete un arreglo con los primeros 100 números enteros a partir del 0 y los muestre en pantalla en orden ascendente.

**EJERCICIO 10.** Escriba un programa que complete un arreglo con los números pares que se encuentren entre 100 y 200 y los muestre en pantalla en orden descendente.

**EJERCICIO 11.** Escriba un programa que complete un arreglo con los primeros 50 múltiplos de 3 y los muestre en pantalla en orden descendente.

**EJERCICIO 12.** Escriba un programa que lea un arreglo  $a$  de 10 enteros y un entero  $n$  y que imprima el índice del arreglo  $a$  donde se encuentra  $n$  si está presente en el arreglo y, -1 en caso contrario.


**EJERCICIO 13.** Escriba un programa que lea un entero  $n$  entre 5 y 100 y luego solicite al usuario el ingreso de  $n$  enteros, los guarde a todos en un arreglo, y finalmente determine si la suma de los elementos del arreglo es mayor a 30. Si el usuario ingresa un número  $n$  menor a 5 o mayor a 100 entonces se deberá imprimir un mensaje de Error y el ingreso del arreglo y el análisis de su contenido no se realizará.

**EJERCICIO 14.** Escriba un programa que lea enteros hasta que se ingrese un número negativo y posteriormente imprimir qué valor entre 0 y 99 se ingresó más veces.

**EJERCICIO 15.** Escriba una función *sumaArr* que tome un arreglo de enteros junto con la longitud del mismo y devuelva la suma de sus elementos.

**EJERCICIO 16.** Escriba una función *sumaAlt* que tome un arreglo de enteros junto con la longitud del mismo y devuelva el producto de los elementos cuyos índices son pares.

**EJERCICIO 17.** Analice el siguiente código e introduzca comentarios en cada línea indicando el objeto de la misma. Observar la 4ta línea a modo de ejemplo. El algoritmo expuesto data del siglo III a.C. y se lo denomina **Criba de Eratóstenes**. ¿Puede entender qué hace el programa?



```
#include <stdio.h>
#define N = 1000;
int main() {
    int i, j, a[N+1]; // línea 4: declaracion de dos enteros y
                      // un arreglo de (completar) componentes
    for (a[1]=0, i=2; i <= N; i++) {
        a[i] = 1;
    }
    for (i=2; i <= N/2; i++) {
        for (j=i*i; j <= N; j+=i) {
            a[j] = 0;
        }
    }
    for (i=1; i <= N; i++) {
        if(a[i]==1) {
            printf("%d", i);
        }
    }
    printf("\n");
    return 0;
}
```

## 4. Cadenas de caracteres

**EJERCICIO 18.** Escriba una función que reciba una cadena de caracteres y un caracter y devuelva 0 si el caracter esta presente en la cadena y 1 en caso contrario.

**EJERCICIO 19.** Escriba una función que reciba una cadena de caracteres y un caracter y devuelva la cantidad de apariciones del caracter en la cadena.

**EJERCICIO 20.** Escriba un programa que lea una cadena de caracteres y la imprima al revés.

**EJERCICIO 21.** Escriba una función que reciba una cadena de caracteres y determine si la misma es capicua.

**EJERCICIO 22.** Escriba una función que reciba una cadena de caracteres y determine si es un pangrama, es decir, si para escribir la línea se utilizaron todos los caracteres del alfabeto.

**EJERCICIO 23.** Escriba una función que reciba dos cadenas de caracteres y determine si una está contenida en la otra.