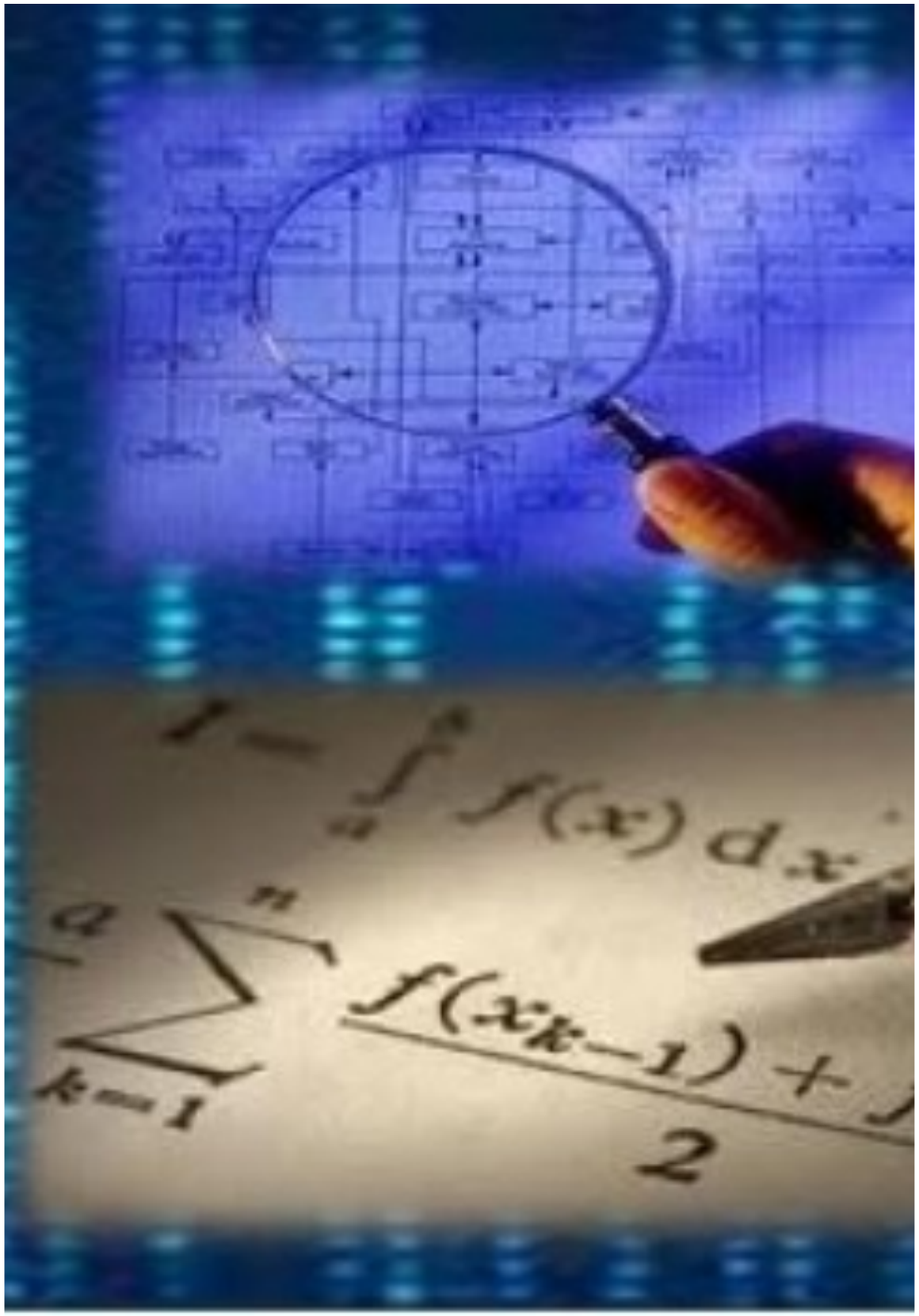
Programación básica

Alma González
Agosto 2021



INSTRUCCIÓN:
WHILE

- La instrucción while nos permite realizar una serie de instrucciones de forma repetida mientras se siga cumpliendo la condición especificada. Es decir nos permite hacer un ciclo. La sintaxis es:

```
while(condicion){

}
```

La condición usualmente es un comparativo entre una variable y un valor de referencia, o bien entre dos variables. La comparación puede ser del tipo: ==,=!,<,>,<=,>=,

Ejemplo 1: While

```c
#include<stdio.h>

int main()
{
    float temp_C,temp_K;
    float inicial=100,final=200,delta;
    int n=10;

    delta=(final-inicial)/n;
    temp_C=inicial;

    while(temp_C<=final){
        temp_K=temp_C+273.15;
        printf("%f %f\n",temp_C,temp_K);
        temp_C=temp_C+delta;        //   temp_C+=delta;

    }
    return(0);
}
```

Ejemplo 2: While

```c
#include<stdio.h>

int main()
{
    float temp_C,temp_K;
    float inicial=100,final=200,delta;
    int n=10;
    int op=1;

    delta=(final-inicial)/n;
    while(op==1){
        temp_K=0.;
        temp_C=inicial;

    while(temp_C<=final){
        temp_K=temp_C+273.15;
        printf("%f %f\n",temp_C,temp_K);
        temp_C=temp_C+delta;        //   temp_C+=delta;

    }
        printf("Deseas hacer otra operacion? Presiona 1 para si, Presiona 2 para no\n");
        scanf("%i",&op);
    }
    return(0);
}
```

Ejemplo 3: While.

```c
#include<stdio.h>

int main()
{
    float temp_C,temp_K;
    float inicial=100,final=200,delta;
    int n=10;
    char op[2];

    op[0] = 's';
    delta=(final-inicial)/n;
    while(op[0]=='s'){
        temp_K=0.;
        temp_C=inicial;

    while(temp_C<=final){
        temp_K=temp_C+273.15;
        printf("%f %f\n",temp_C,temp_K);
        temp_C=temp_C+delta;        //   temp_C+=delta;

    }
        printf("Deseas hacer otra operacion? (si/no)\n");
        scanf("%s",op);
    }
    return(0);
}
```

INSTRUCCIÓN:
FOR

- La instrucción for nos permite realizar una serie de instrucciones de forma repetida por un determinado numero de veces, hace un ciclo. Está instrucción requiere de llevar un contador que indique el numero de veces que se han realizado las instrucciones. Cuando el contador llega a un numero máximo pre-definido el ciclo termina. La sintaxis es:

```
for(i=0; i<=n_iter; i++){

Instrucciones

}
```

Instrucción:

For

```
for(i=0; i<n_iter; i++ ) {
}
```

i: contador, variable entera (declarar variable)
puede ser cualquier otro nombre de variable.
ej: i,j,k,cont, num, etc…

n_iter: variable entera, (declarar variable). Condición a checar para seguir realizando el ciclo, si la condición no se cumple el ciclo termina.

i++: la variable i aumenta en 1 unidad cada vez que se completa la serie de instrucciones.

EJEMPLO 1: FOR

```c
#include<stdio.h>

int main()
{
    float temp_C,temp_K;
    float Temp_C=100,final=200,delta;
    int n=10,i;

    delta=(final-Temp_C)/n;
   for(i=0;i<n;i++){
        temp_K=temp_C+273.15;
        printf("%f %f\n",temp_C,temp_K);
        temp_C=temp_C+delta;        //   temp_C+=delta;

    }
    return(0);
}
```

Ejercicio, añade una instrucción while para que el usuario pueda pedir ejecutar el programa nuevamente. Y haz que el usuario pueda definir las temperaturas inicial (Temp_C), final (final) y numero de pasos (n).

EJEMPLO 1: FOR

```c
#include<stdio.h>
#include<math.h>

int main()
{
    float exp_;
   int n=10,j;

    for(j=0;j<n;j++){
        exp_=exp(j);
        printf("%i \t %f\n",j,exp_);
     }
    return 0;
}
```

EJEMPLO 1: FOR

ES POSIBLE HACER CICLOS FOR ANIDADOS (UNO DENTRO DE OTRO)

```c
#include<stdio.h>
#include<math.h>

int main()
{
 float res;

 int n=3,k,j;

    for(j=0;j<n;j++){
        for(k=0;k<n;k++){
        res=j*k*1.0;
        printf("%i %i %f\n",j,k,res);
         }
    }
    return 0;
}
```

EJERCICIOS:

- RECUERDEN QUE TODOS LOS PROGRAMAS SE DEBE CONTAR CON UN ALGORITMO/DIAGRAMA.

- Haz un programa que evalúe las funciones: exponencial (exp), logaritmo (log), seno (sin), coseno (cos), y raíz cuadrada (sqrt) de una variable x, en un intervalo y con un espaciado definido por el usuario. El nombre del programa debe ser funciones.c

- Haz un programa que encuentre, e imprima en la pantalla, todos los números primos en un intervalo definido por el usuario. Ej. Entre 3 y 10 hay 3 números primos 3, 5, 7. Asegurate que el intervalo no tenga más de 100 numero primos. Imprime los números a la pantalla en renglones de 20 números máximo. Es decir que máximo se impriman 5 lineas a la pantalla. Si no es el caso pide al usuario que de un nuevo intervalo. El nombre del programa debe ser numeros_primos.c