	<b>Carátula para entrega de prácticas</b>	
<b>Facultad de Ingeniería</b>		<b>Laboratorios de docencia</b>

**Laboratorio de Computación**

**Salas A y B**

<b>Profesor(a):</b>	<b>César Fabián Domínguez Velasco</b>
<b>Asignatura:</b>	<b>Fundamentos de Programación</b>
<b>Grupo:</b>	<b>15</b>
<b>No de Práctica(s):</b>	<b>8</b>
<b>Integrante(s):</b>	<b>Camacho Duarte Héctor Enrique</b>
	<b>Gutiérrez Esquivel Giovani Emiliano</b>
	<b>Flores Jiménez Diego</b>
	<b>Moreno Chapan Amilet</b>
	<b>Reyes García Raúl de Jesús</b>
<b>No. de lista</b>	<b>07,11,15,16,25,35.</b>
<b>o brigada:</b>	
<b>Semestre:</b>	<b>2024-02</b>
<b>Fecha de entrega:</b>	<b>10/04/24</b>
<b>Observaciones:</b>	
<b>CALIFICACIÓN:</b>	

## **Introducción:**

Las estructuras de repetición, a menudo denominadas como ciclos o bucles, son pilares fundamentales en el desarrollo de software. Estas herramientas permiten la ejecución iterativa de un conjunto de instrucciones mientras se cumpla una condición lógica específica.

En el contexto del lenguaje C, se destacan tres tipos principales de estructuras de repetición que son las que checamos en la práctica:

**While:** Esta estructura ejecuta un bloque de código mientras una expresión lógica sea verdadera.

**Do-While:** Similar al while, el do-while repite un bloque de código, asegurando al menos una ejecución incluso si la condición inicial es falsa.

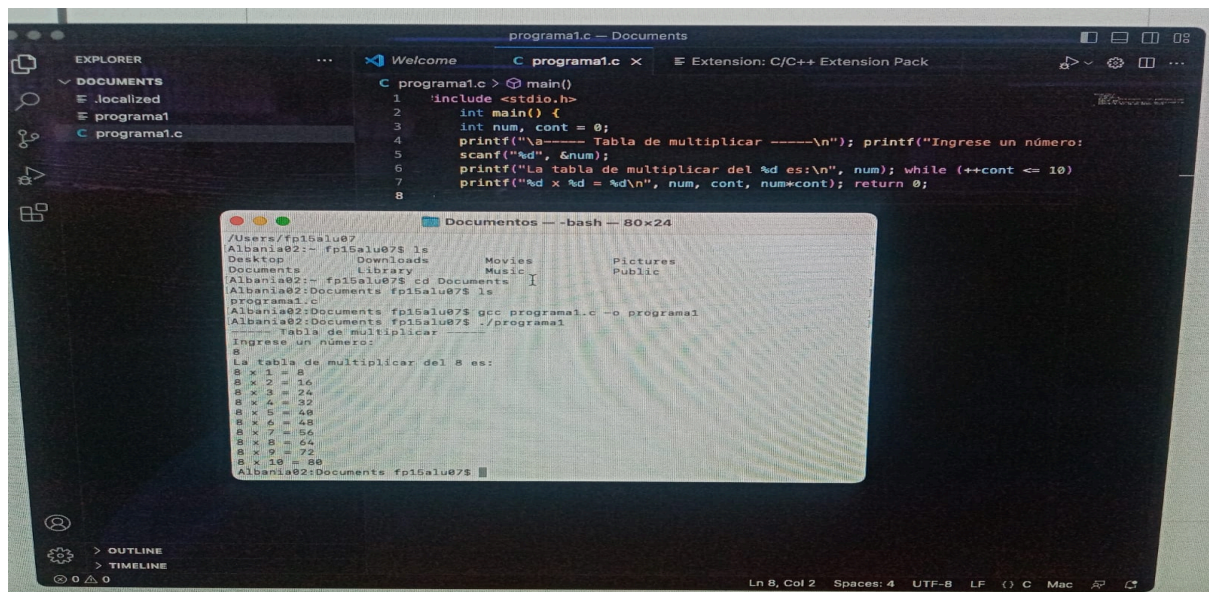
**For:** Especialmente útil para iteraciones controladas, la estructura for permite definir una variable de control, una condición de terminación y una actualización en una sola línea.

Estas estructuras son cruciales para la creación de programas eficientes y funcionales. La elección adecuada entre ellas depende de la lógica específica de la aplicación y los requisitos del problema a resolver a continuación ejecutaremos los ejercicios de la práctica y el ejercicio propuesto.

## **Objetivo:**

El alumno elaborará programas en C para la resolución de problemas básicos que incluyan las estructuras de repetición.

## Programa1.c



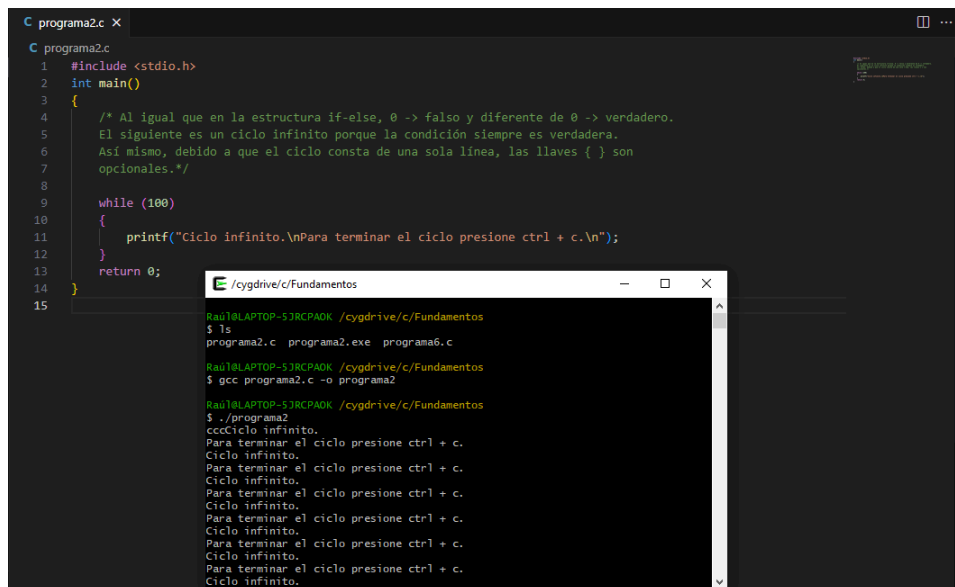
Este programa calcula y muestra la tabla de multiplicar del número ingresado por el usuario utilizando un bucle con el ciclo while.

## Programa2.c

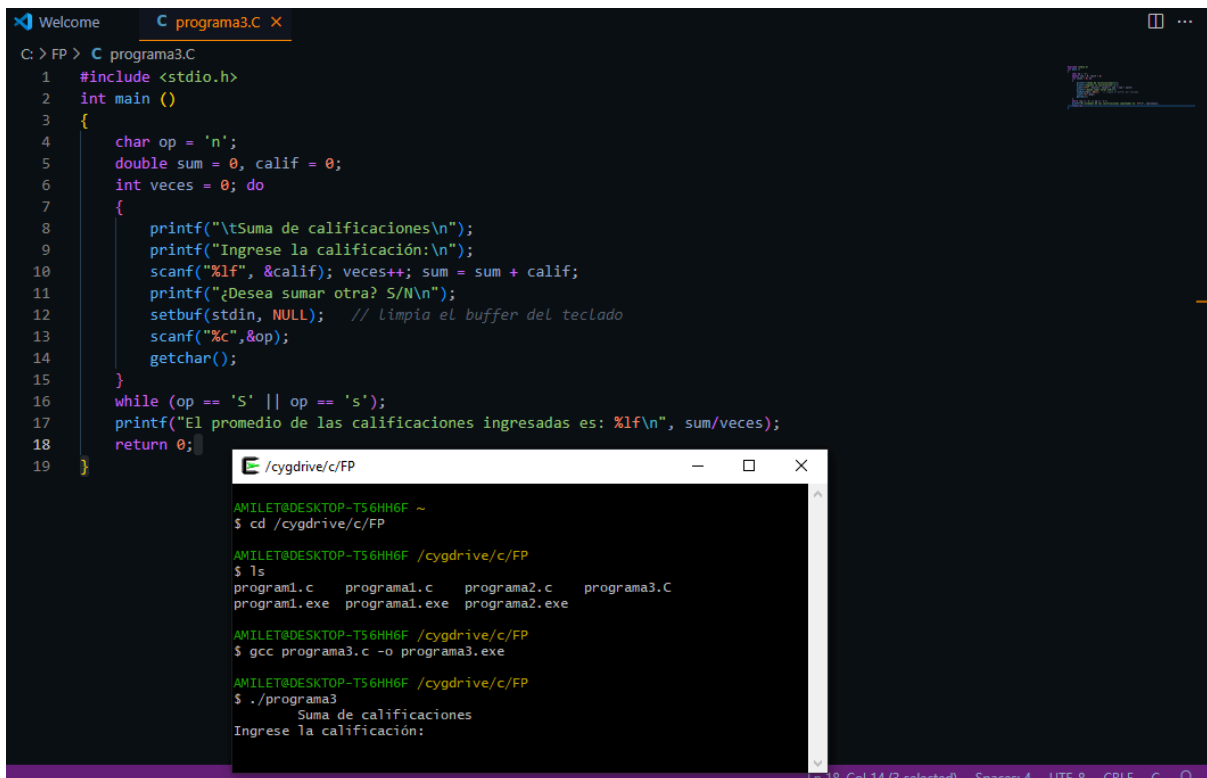
En este programa, haciendo el uso del ciclo while se realizó un bucle infinito que mostrará el mensaje

## “Ciclo infinito

Para terminar el ciclo presione ctrl + c" hasta que el usuario quiera detenerlo" el bucle no parará hasta que se introduzca la combinación ctrl+c.



## Programa3.c



```
1 #include <stdio.h>
2 int main ()
3 {
4     char op = 'n';
5     double sum = 0, calif = 0;
6     int veces = 0; do
7     {
8         printf("\tSuma de calificaciones\n");
9         printf("Ingrese la calificación:\n");
10        scanf("%lf", &calif); veces++; sum = sum + calif;
11        printf("¿Desea sumar otra? S/N\n");
12        setbuf(stdin, NULL); // limpia el buffer del teclado
13        scanf("%c",&op);
14        getchar();
15    }
16    while (op == 's' || op == 'S');
17    printf("El promedio de las calificaciones ingresadas es: %lf\n", sum/veces);
18    return 0;
19 }
```

```
AMILET@DESKTOP-T56HH6F ~
$ cd /cygdrive/c/FP

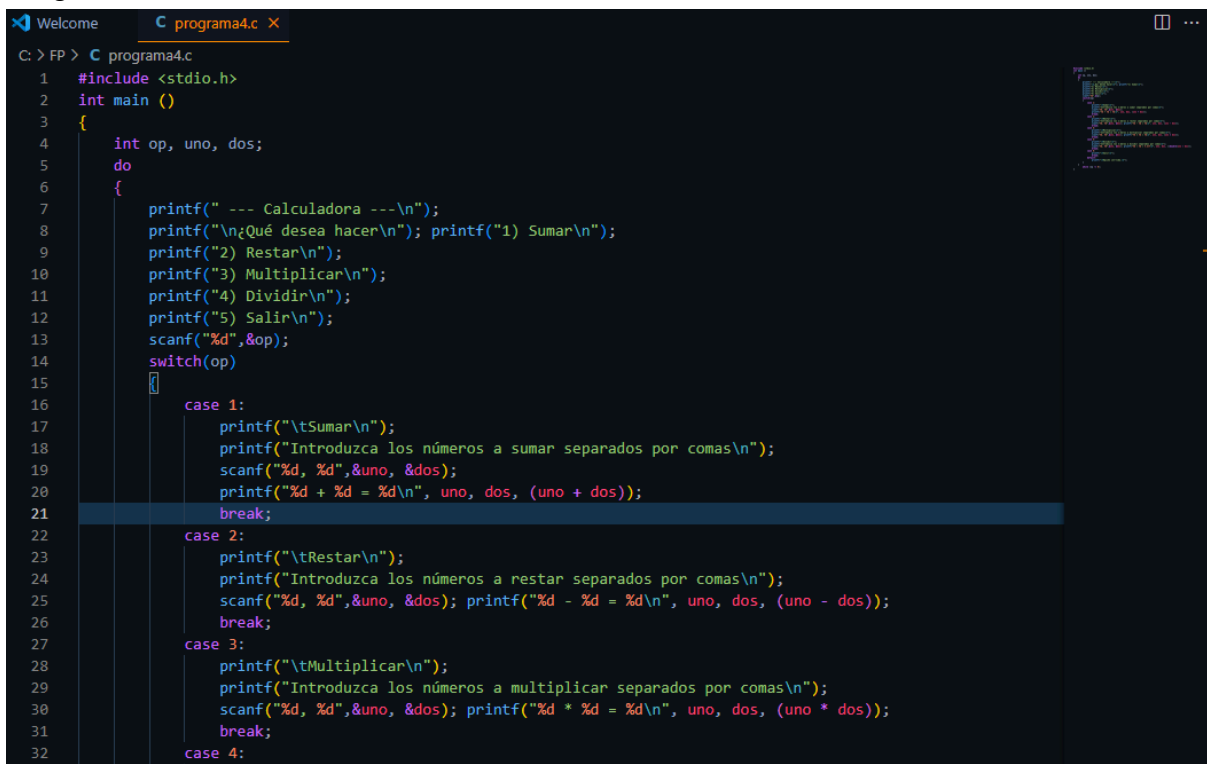
AMILET@DESKTOP-T56HH6F /cygdrive/c/FP
$ ls
programa1.c  programa1.c  programa2.c  programa3.C
programa1.exe  programa1.exe  programa2.exe

AMILET@DESKTOP-T56HH6F /cygdrive/c/FP
$ gcc programa3.c -o programa3.exe

AMILET@DESKTOP-T56HH6F /cygdrive/c/FP
$ ./programa3
Suma de calificaciones
Ingrese la calificación:
```

En este programa se calcula el promedio de una persona X, se solicitan las calificaciones que posteriormente serán sumadas y divididas entre el número de calificaciones ingresadas, haciendo el uso del ciclo Do - While, mostrando al final el promedio obtenido

## Programa4.c



```
1 #include <stdio.h>
2 int main ()
3 {
4     int op, uno, dos;
5     do
6     {
7         printf(" --- Calculadora ---\n");
8         printf("\n¿Qué desea hacer\n"); printf("1) Sumar\n");
9         printf("2) Restar\n");
10        printf("3) Multiplicar\n");
11        printf("4) Dividir\n");
12        printf("5) Salir\n");
13        scanf("%d",&op);
14        switch(op)
15        {
16            case 1:
17                printf("\tSumar\n");
18                printf("Introduzca los números a sumar separados por comas\n");
19                scanf("%d, %d",&uno, &dos);
20                printf("%d + %d = %d\n", uno, dos, (uno + dos));
21                break;
22            case 2:
23                printf("\tRestar\n");
24                printf("Introduzca los números a restar separados por comas\n");
25                scanf("%d, %d",&uno, &dos); printf("%d - %d = %d\n", uno, dos, (uno - dos));
26                break;
27            case 3:
28                printf("\tMultiplicar\n");
29                printf("Introduzca los números a multiplicar separados por comas\n");
30                scanf("%d, %d",&uno, &dos); printf("%d * %d = %d\n", uno, dos, (uno * dos));
31                break;
32            case 4:
```

```
Welcome  C programa4.c X
C: > FP > C programa4.c
3  {
6  {
15 {
16     case 1:
21         break;
22     case 2:
23         printf("\tRestar\n");
24         printf("Introduzca los números a restar separados por comas\n");
25         scanf("%d, %d", &uno, &dos); printf("%d - %d = %d\n", uno, dos, (uno - dos));
26         break;
27     case 3:
28         printf("\tMultiplicar\n");
29         printf("Introduzca los números a multiplicar separados por comas\n");
30         scanf("%d, %d", &uno, &dos); printf("%d * %d = %d\n", uno, dos, (uno * dos));
31         break;
32     case 4:
33         printf("\tDividir\n");
34         printf("Introduzca los números a dividir separados por comas\n");
35         scanf("%d, %d", &uno, &dos); printf("%d / %d = %.2lf\n", uno, dos, ((double)uno / dos));
36         break;
37     case 5:
38         printf("\tSalir\n");
39         break;
40     default:
41         printf("\tOpción inválida.\n");
42     }
43 }
44 while (op != 5);
45 }
```

```
/cygdrive/c/FP
AMILET@DESKTOP-T56HH6F ~
$ cd /cygdrive/c/FP

AMILET@DESKTOP-T56HH6F /cygdrive/c/FP
$ LS
program1.c  programa1.c  programa2.c  programa3.C  programa4.c
program1.exe programa1.exe programa2.exe programa3.exe

AMILET@DESKTOP-T56HH6F /cygdrive/c/FP
$ gcc programa4.c -o programa4.exe

AMILET@DESKTOP-T56HH6F /cygdrive/c/FP
$ ./programa4
--- Calculadora ---

¿Qué desea hacer
1) Sumar
2) Restar
3) Multiplicar
4) Dividir
5) Salir
```

En este programa se realizó un menú para una Calculadora que esta haga suma, resta, división y multiplicación de dos datos, para esto se hizo uso de un ciclo Do - While para poder repetir dicho menú, una estructura de selección (switch) para que el usuario ingrese a la opción de su interés ingresando el número o carácter en dicho menú especificado

## Programa5.c

```
C programa5.c x
C programa5.c
1  #include <stdio.h>
2  #define MAX 5
3  int main ()
4  {
5      int arreglo[MAX], cont;
6      for (cont=0; cont<MAX; cont++)
7      {
8          printf("Ingrese el valor %d del arreglo: ", cont+1);
9          scanf("%i", &arreglo[cont]);
10     }
11
12     printf("El valor ingresado para cada elemento del arreglo es:\n");
13     for (cont=0; cont<MAX; cont++)
14     {
15         printf("%d\t", arreglo[cont]);
16     }
17     printf("\n");
18     return 0;
19 }
20
```

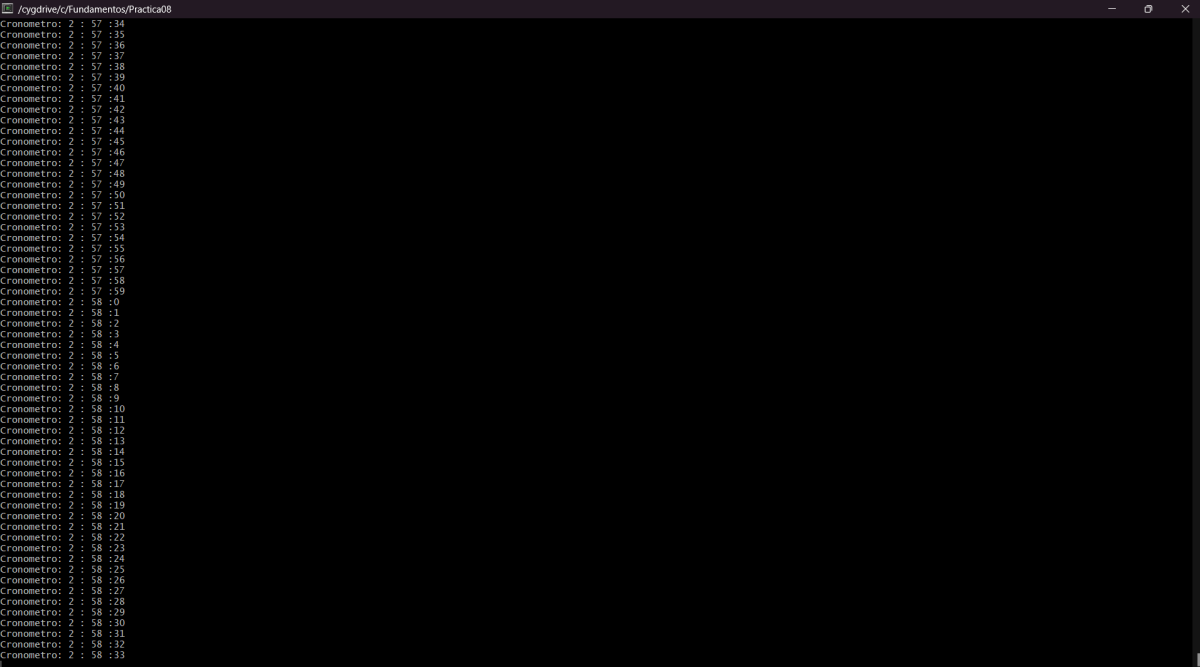
```
/cygdrive/c/Fundamentos
Raúl@LAPTOP-5JRCPAOK /cygdrive/c/Fundamentos
$ ls
programa2.c programa2.exe programa5.c programa6.c
Raúl@LAPTOP-5JRCPAOK /cygdrive/c/Fundamentos
$ gcc programa5.c -o programa5
Raúl@LAPTOP-5JRCPAOK /cygdrive/c/Fundamentos
$ ./programa5
Ingrese el valor 1 del arreglo: 7
Ingrese el valor 2 del arreglo: 9
Ingrese el valor 3 del arreglo: 0
Ingrese el valor 4 del arreglo: 4
Ingrese el valor 5 del arreglo: 3
El valor ingresado para cada elemento del arreglo es:
[7    9    0    4    3    ]
Raúl@LAPTOP-5JRCPAOK /cygdrive/c/Fundamentos
$ |
```

Este programa solicita al usuario cierto número de datos (5 datos) para mostrarlos en un array haciendo el uso de un ciclo for, seguido usando un nuevo ciclo for para mostrar dicho array

## Problemas extras elaborados en el laboratorio:

### 1. Elaborar un cronómetro con estructura de repetición for

```
Contador.c > ...
1  ✓ #include <stdio.h>
2    #include <windows.h>
3
4  ✓ int main(int argc, char const *argv[])
5  {
6      int horas, minutos, segundos;
7
8      for (horas = 0; horas < 24; horas++)
9      {
10         for (minutos = 0; minutos <= 59; minutos++)
11         {
12             for (segundos = 0; segundos <= 59; segundos++)
13             {
14                 printf("Cronometro: %d : %d :%d \n", horas, minutos, segundos);
15                 sleep(1);
16             }
17         }
18     }
19
20 }
21
22 return 0;
23 }
24
```



Haciendo uso de 3 ciclos for se elaboró un cronómetro cada for controla cada parámetro de nuestro cronometro, un for para los horas, otro para los minutos y uno para los segundos

## 2. Adivinar el un número aleatorio

```
C Juego.c > main(int, char const * [])
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <time.h>
4
5  int main(int argc, char const *argv[])
6  {
7      int random;
8      int opc;
9
10     printf("--- Adivinar el numero ---\n");
11
12     srand(time(NULL));
13     random = rand()%11;
14     do
15     {
16         printf("Dame un número del 0 al 10\n");
17         scanf("%d", &opc);
18
19         if (opc == random)
20         {
21             printf("Felicidades encontraste el numero C: \n");
22         }
23         else
24         {
25             printf("Sigue intentando :C\n");
26         }
27     } while (random != opc);
28
29     return 0;
30 }
31
```



```
/cygdrive/c/Fundamentos/Practica08

ineff@Inefable /cygdrive/c/Fundamentos/Practica08
$ gcc Juego.c -o Juego.exe

ineff@Inefable /cygdrive/c/Fundamentos/Practica08
$ ./Juego
--- Adivinar el numero ---
numero generado: 2
Dame un número del 0 al 10
ineff@Inefable /cygdrive/c/Fundamentos/Practica08
$ gcc Juego.c -o Juego.exe

ineff@Inefable /cygdrive/c/Fundamentos/Practica08
$ ./Juego
--- Adivinar el numero ---
Dame un número del 0 al 10
0
Sigue intentando :C
Dame un número del 0 al 10
1
Sigue intentando :C
Dame un número del 0 al 10
2
Sigue intentando :C
Dame un número del 0 al 10
3
Sigue intentando :C
Dame un número del 0 al 10
4
Sigue intentando :C
Dame un número del 0 al 10
5
Sigue intentando :C
Dame un número del 0 al 10
6
Sigue intentando :C
Dame un número del 0 al 10
7
Sigue intentando :C
Dame un número del 0 al 10
8
Felicidades encontraste el numero C:

ineff@Inefable /cygdrive/c/Fundamentos/Practica08
$
```

En este programa usando un ciclo Do - While para controlar el número de veces que queramos hacer intentos para adivinar nuestro número y srand para “generar” un número aleatorio en un rango definido, una vez capturado el número por el usuario este se evaluará en una estructura condicional if- else si este es correcto dara un mensaje felicitando al usuario por encontrarlo, por el otro lado un mensaje para que siga intentando

## Ejercicio propuesto

```
#include <stdio.h>

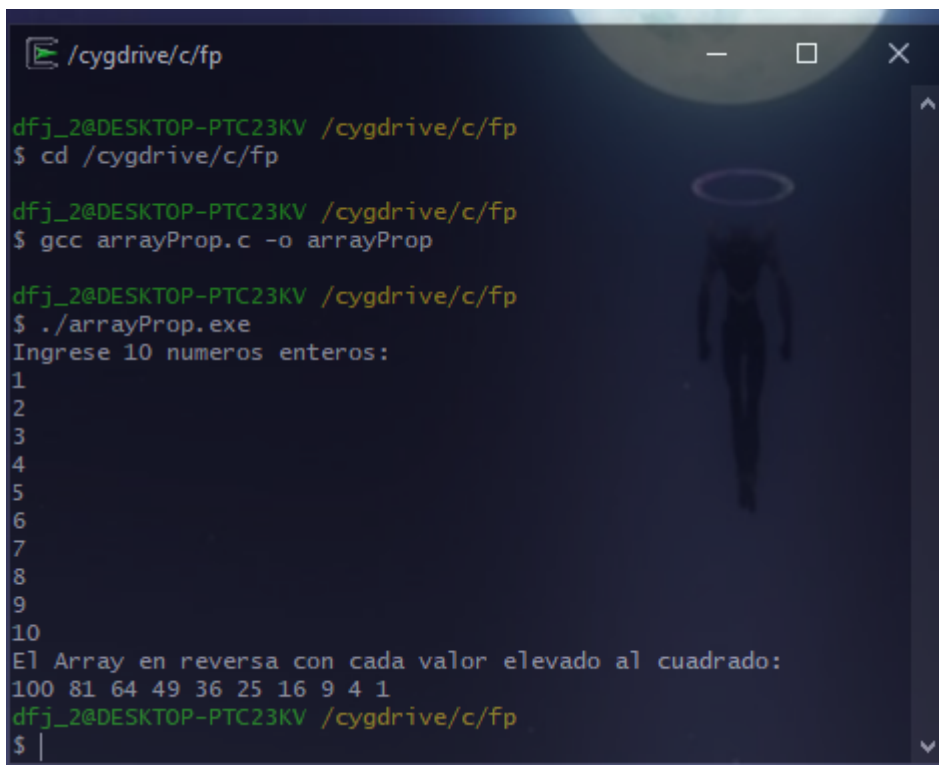
#define SIZE_ARRAY 10

int main() {
    int NUM_ARRAY[SIZE_ARRAY];
    int i;

    printf("Ingrese %d numeros enteros:\n", SIZE_ARRAY);
    for (i = 0; i < SIZE_ARRAY; i++) {
        scanf("%d", &NUM_ARRAY[i]);
    }

    printf("El Array en reversa con cada valor elevado al cuadrado:\n");
    for (i = SIZE_ARRAY - 1; i >= 0; i--) {
        printf("%d ", NUM_ARRAY[i] * NUM_ARRAY[i]);
    }

    return 0;
}
```



```
/cygdrive/c/fp
dfj_2@DESKTOP-PTC23KV /cygdrive/c/fp
$ cd /cygdrive/c/fp
dfj_2@DESKTOP-PTC23KV /cygdrive/c/fp
$ gcc arrayProp.c -o arrayProp
dfj_2@DESKTOP-PTC23KV /cygdrive/c/fp
$ ./arrayProp.exe
Ingrese 10 numeros enteros:
1
2
3
4
5
6
7
8
9
10
El Array en reversa con cada valor elevado al cuadrado:
100 81 64 49 36 25 16 9 4 1
dfj_2@DESKTOP-PTC23KV /cygdrive/c/fp
$ |
```

En este programa haciendo el uso de ciclos for, se inicializa un array numérico de un tamaño definido (10 números enteros), que estos serán elevados al cuadrado y posteriormente imprimir estos de manera inversa a la que fueron ingresados dichos números

## **Conclusión**

Cumplimos el objetivo propuesto, ya que ejecutamos y elaboramos programas en C para la resolución de problemas básicos que incluyen las estructuras de repetición while, do while y for, adicionalmente retomamos la estructuras de selección como lo es el if, else y switch.

Además, revisamos la importancia de optimizar un programa, quizá podamos llegar a el mismo resultado con alguna otra estructura de repetición pero es importante analizar cuál estructura es la que mejor se adecua a la resolución de nuestro problema.

Conocimos la directiva “#define” que nos permite asignar valores como a variables “a” o palabras como “tamaño” y en muchas ocasiones puede ser de utilidad ya que luego, cada vez que se utiliza ese nombre en el código, el compilador lo reemplaza por el valor o la secuencia de código definidos.

Finalmente el profesor Yovanninos mostró cómo utilizando estructuras de repetición podemos elaborar, juegos, relojes o contadores.

## **Bibliografía**

*Laboratorio Salas A y B.* (n.d.). <http://lcp02.fi-b.unam.mx/>

*El lenguaje de programación C.* Brian W. Kernighan, Dennis M. Ritchie, segunda edición, USA, Pearson Educación 1991.