



**INSTITUTO POLITÉCNICO NACIONAL**



**Escuela Superior de Cómputo**

**Estructuras de Datos**

**Nombre:**

Macias Castillo Josue

**Profesor:**

Edgardo Adrián Franco Martínez

**Grupo:**

1CM9

**Boleta:**

2015301058

**Problema:**

Programa obtener suma y resta de 2 matrices

## Índice

**Índice .....2**

**Implementación ..... 3**

**Pruebas ..... 6**

## Implementación

Se tiene que crear el código del siguiente problema:

Programa de manera estructurada en ANSI C, un programa capaz de generar y recibir dos matrices dinámicas de tamaño  $m$  por  $n$ , donde  $m$  y  $n$  ingresan por la entrada estándar, además de los valores de las matrices, el programa realiza una operación de suma y resta de matrices que se muestra por la salida estándar.

A partir del problema que se nos proporcionó se pasó a la implementación del mismo:

```

1  /*
2  AUTOR: Josue Macias Castillo (C) Agosto 2016
3  VERSIÓN: 1.2
4
5  DESCRIPCIÓN: Programa solución al problema: "Se necesita realizar un programa que sea capaz de recibir y generar 2 matrices dinámicas de tamaño m x n,
6  el program realiza una operación de suma y resta de matrices que se muestra por la salida estándar".
7
8  OBSERVACIONES:
9  El programa recibe en el programa principal todas las funciones que generan las matrices y los datos por medio de la entrada estándar para saber
10 de cuantas columnas y filas es cada matriz
11
12 COMPILACIÓN: gcc programa.c -o programa
13 EJECUCIÓN: programa.exe (En Windows) - ./programa (En Linux)
14 */
15
16 //LIBRERIAS
17 #include<stdio.h> //Incluye las funciones estandar de entrada y salida
18 #include<conio.h> //Incluye el getch para que el programa al ejecutarse no se cierre solo
19 #include<stdlib.h> //Incluye las funciones de asignación dinámica
20
21 #define Limpiar system("cls") //Define system("cls") como limpiar para hacerlo mas practico en el codigo
22
23 //DECLARACIÓN DE FUNCIONES
24 //Aparta el espacio de memoria para la primer matriz dinámica
25 int **Aparta_Matriz(int m, int n);
26 //Aparta el espacio de memoria para la segunda matriz dinámica
27 int **Aparta_Matriz_2(int m, int n);
28 //Captura la primer matriz dinámica
29 void Capturar_Matriz(int **matriz, int m, int n);
30 //Captura la segunda matriz dinámica
31 void Capturar_Matriz_2(int **matriz_2, int m, int n);
32 //Imprime la primer matriz dinámica
33 void Imprime_Matriz(int **matriz, int m, int n);
34 //Imprime la segunda matriz dinámica
35 void Imprime_Matriz_2(int **matriz_2, int m, int n);
36 //Suma las matrices
37 void Suma_Matrices(int **matriz, int **matriz_2, int m, int n);
38 //Resta las matrices
39 void Resta_Matrices(int **matriz, int **matriz_2, int m, int n);
40
41 //PROGRAMA PRINCIPAL
42 int main(void)
43 {
44     int i, j, m, n, **matriz, **matriz_2, Suma, Resta;
45     //Leer m y n
46     printf("Introduce m y n separados por espacio: [   ]\b\b\b\b");
47     scanf("%d %d",&m,&n);
48     //Aparta memoria para la primer matriz dinámica
49     matriz = Aparta_Matriz(m,n);
50     //Captura la primer matriz dinámica
51     Capturar_Matriz(matriz,m,n);
52     //Leer m y n para la segunda matriz
53     printf("Introduce m y n separados por espacio: [   ]\b\b\b\b");
54     scanf("%d %d",&m,&n);
55     //Aparta memoria para la segunda matriz dinámica
56     matriz_2 = Aparta_Matriz_2(m,n);
57     //Captura la segunda matriz dinámica
58     Capturar_Matriz_2(matriz_2,m,n);
59     Limpiar; //Limpia la pantalla para mostrar los resultados
60     //Muestra la primer matriz
61     printf("\n*****Matriz Original 1*****");
62     Imprime_Matriz(matriz, m, n);
63     //Muestra la segunda matriz
64     printf("\n*****Matriz Original 2*****");
65     Imprime_Matriz(matriz_2, m, n);
66     //Muestra la suma de las matrices
67     printf("\n*****Suma de las matrices*****");
68     Suma_Matrices(matriz,matriz_2,m,n);
69

```

```

70 //Muestra la resta de las matrices
71 printf("\n*****Resta de las matrices*****");
72 Resta_Matrices(matriz,matriz_2,m,n);
73
74 getch();
75 }
76
77 /*
78 int **Aparta_Matriz(int m,int n);
79 Descripción: Aparta el espacio de memoria para la primera matriz dinámica
80 Recibe: int m (Número de filas), int n (Número de columnas)
81 Devuelve: int **matriz (Referencia a la matriz apartada),
82 Observaciones: m y n deberan ser mayor a 0
83 */
84 int **Aparta_Matriz(int m, int n)
85 {
86     int i, **matriz;
87     //Aparta memoria para la primera matriz
88     matriz = (int **)malloc(m*sizeof(int)); //m apuntadores simples a enteros, ademas de que se le aplico un cast
89     for(i = 0; i < m;i++)
90     {
91         matriz[i] = (int *)malloc(n*sizeof(int)); //El apuntador simple matriz[i] apunta a n enteros, ademas de que se le aplico un cast
92     }
93     return matriz; //Regresa la referencia al apuntador principal a la matriz dinámica
94 }
95 /*
96 int **Aparta_Matriz_2(int m,int n);
97 Descripción: Aparta el espacio de memoria para la segunda matriz dinámica
98 Recibe: int m (Número de filas), int n (Número de columnas)
99 Devuelve: int **matriz_2 (Referencia a la matriz apartada),
100 Observaciones: m y n deberan ser mayor a 0
101 */
102 int **Aparta_Matriz_2(int m, int n)
103 {
104     int i, **matriz_2;
105     //Aparta memoria para la segunda matriz
106     matriz_2 = (int **)malloc(m*sizeof(int)); //m apuntadores simples a enteros, ademas de que se le aplico un cast
107     for(i = 0; i < m;i++)
108     {
109         matriz_2[i] = (int *)malloc(n*sizeof(int)); //El apuntador simple matriz[i] apunta a n enteros, ademas de que se le aplico un cast
110     }
111     return matriz_2; //Regresa la referencia al apuntador principal a la matriz dinámica
112 }
113 /*
114 void Capturar_Matriz(int **matriz,int m,int n);
115 Descripción: Capturar la primer matriz dinámica
116 Recibe: int **matriz (Referencia a la matriz), int m (Número de filas), int n (Número de columnas)
117 Devuelve:
118 Observaciones: La matriz fue reservada previamente de tamaño m por n
119 */
120 void Capturar_Matriz(int **matriz, int m, int n)
121 {
122     int i, j;
123
124     for(i = 0; i < m; i++)
125     {
126         for(j = 0; j < n; j++)
127         {
128             printf("Introduce el elemento[%d][%d]: ",i,j);
129             scanf("%d",&matriz[i][j]);
130         }
131     }
132 }

```

```

133 /*
134 void Capturar_Matriz_2(int **matriz_2,int m,int n);
135 Descripción: Capturar la segunda matriz dinámica
136 Recibe: int **matriz_2 (Referencia a la matriz), int m (Número de filas), int n (Número de columnas)
137 Devuelve:
138 Observaciones: La matriz fue reservada previamente de tamaño m por n
139 */
140 void Capturar_Matriz_2(int **matriz_2, int m, int n)
141 {
142     int i, j;
143     for(i = 0; i < m; i++)
144     {
145         for(j = 0; j < n; j++)
146         {
147             printf("Introduce el elemento[%d][%d]: ", i, j);
148             scanf("%d", &matriz_2[i][j]);
149         }
150     }
151 }
152
153 /*
154 void Imprime_Matriz(int **matriz,int m,int n);
155 Descripción: Procedimiento para imprimir la primer matriz
156 Recibe: int **matriz (Referencia a la matriz), int m (Número de filas), int n (Número de columnas)
157 Devuelve:
158 Observaciones: La matriz fue reservada previamente de tamaño m por n
159 */
160 void Imprime_Matriz(int **matriz, int m, int n)
161 {
162     int i, j;
163     for(i = 0; i < m; i++)
164     {
165         printf("\n");
166         for(j = 0; j < n; j++)
167         {
168             printf("\t%d", matriz[i][j]);
169         }
170     }
171 }
172
173 /*
174 void Imprime_Matriz_2(int **matriz_2,int m,int n);
175 Descripción: Procedimiento para imprimir la segunda matriz
176 Recibe: int **matriz_2 (Referencia a la matriz), int m (Número de filas), int n (Número de columnas)
177 Devuelve:
178 Observaciones: La matriz fue reservada previamente de tamaño m por n
179 */
180 void Imprime_Matriz_2(int **matriz_2, int m, int n)
181 {
182     int i, j;
183     for(i = 0; i < m; i++)
184     {
185         printf("\n");
186         for(j = 0; j < n; j++)
187         {
188             printf("\t%d", matriz_2[i][j]);
189         }
190     }
191 }
192

```

```

193 /*
194 void Suma_Matrices(int **Matriz, int **Matriz_2, int Suma, int m, int n)
195 Descripción: Procedimiento para sumar las 2 matrices
196 Recibe: int **matriz (Referencia a la primer matriz), int **matriz_2 (Referencia a la segunda matriz),
197 int m (Número de filas), int n (Número de columnas), int suma (Guarda el resultado)
198 Devuelve: El resultado de la suma
199 Observaciones: La matriz fue reservada previamente de tamaño m por n
200 */
201 void Suma_Matrices(int **matriz, int **matriz_2, int m, int n)
202 {
203     int i, j;
204     for(i = 0; i < m; i++)
205     {
206         printf("\n");
207         for(j = 0; j < n; j++)
208         {
209             printf("\t%d", matriz[i][j] + matriz_2[i][j]); //Aquí se hace la suma de las matrices
210         }
211     }
212 }
213
214 /*
215 void Resta_Matrices(int **matriz, int **matriz_2, int m, int n)
216 Descripción: Procedimiento para sumar las 2 matrices
217 Recibe: int **matriz (Referencia a la primer matriz), int **matriz_2 (Referencia a la segunda matriz),
218 int m (Número de filas), int n (Número de columnas), int suma (Guarda el resultado)
219 Devuelve: El resultado de la suma
220 Observaciones: La matriz fue reservada previamente de tamaño m por n
221 */
222 void Resta_Matrices(int **matriz, int **matriz_2, int m, int n)
223 {
224     int i, j;
225     for(i = 0; i < m; i++)
226     {
227         printf("\n");
228         for(j = 0; j < n; j++)
229         {
230             printf("\t%d", matriz[i][j] - matriz_2[i][j]); //Aquí se hace la resta de las matrices
231         }
232     }
233 }
234
235

```

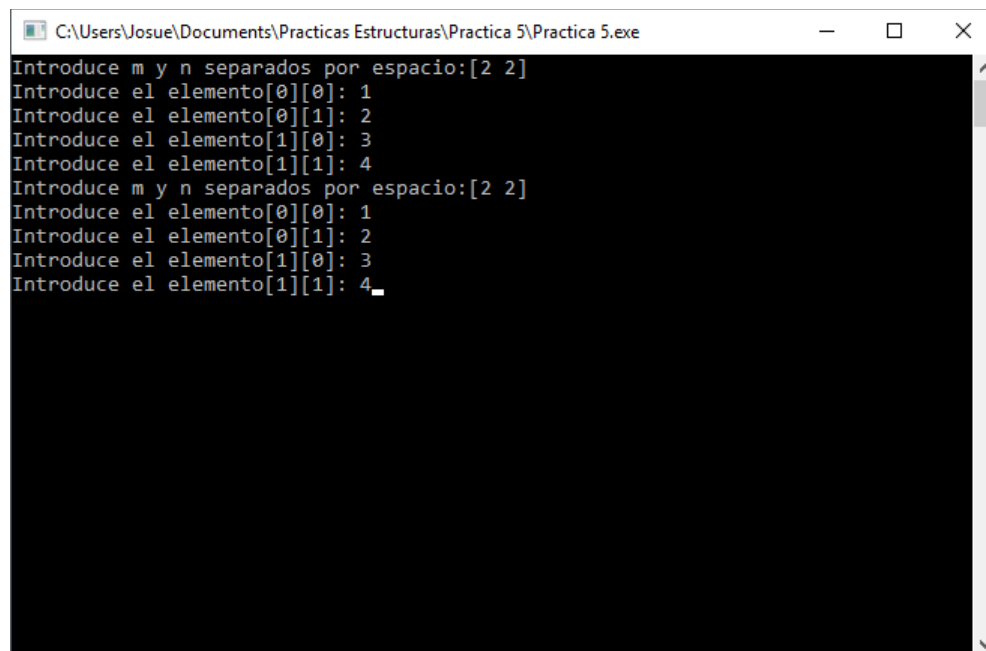
Como se observa en las capturas de pantalla anteriores se hizo una programación modular para poder resolver este ejercicio.

## Pruebas:

Después de ejecutar el programa se hicieron 3 pruebas con diferentes tamaños de matrices: 2x2 y 3x3, gracias a la implementación de memoria dinámica el programa puede tomar cualquier tamaño de matriz ya que no tiene una memoria fija.

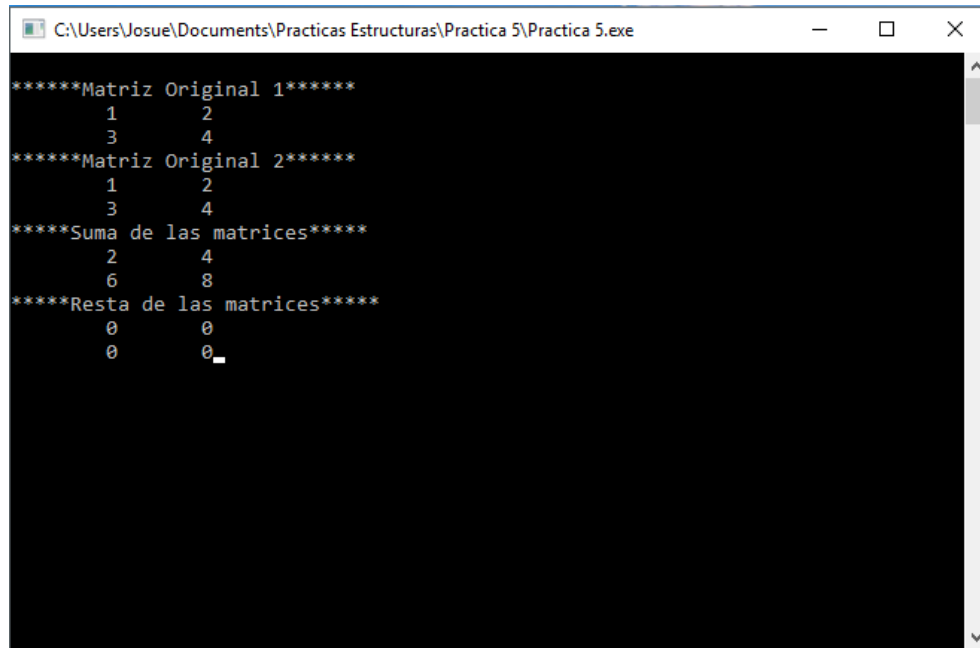
### Ejemplo matriz 2x2:

En la captura se muestra como el usuario dio un tamaño 2x2 y el programa le pidió los valores que esta contendría al finalizar volvía a preguntar para agregar la segunda matriz.



```
C:\Users\Josue\Documents\Practicas Estructuras\Practica 5\Practica 5.exe
Introduce m y n separados por espacio:[2 2]
Introduce el elemento[0][0]: 1
Introduce el elemento[0][1]: 2
Introduce el elemento[1][0]: 3
Introduce el elemento[1][1]: 4
Introduce m y n separados por espacio:[2 2]
Introduce el elemento[0][0]: 1
Introduce el elemento[0][1]: 2
Introduce el elemento[1][0]: 3
Introduce el elemento[1][1]: 4_
```

Cuando el usuario ingresaba todos los valores el programa hace una limpieza de pantalla y muestra las matrices que ingreso de forma ordenada, además de la suma y resta de ambas.

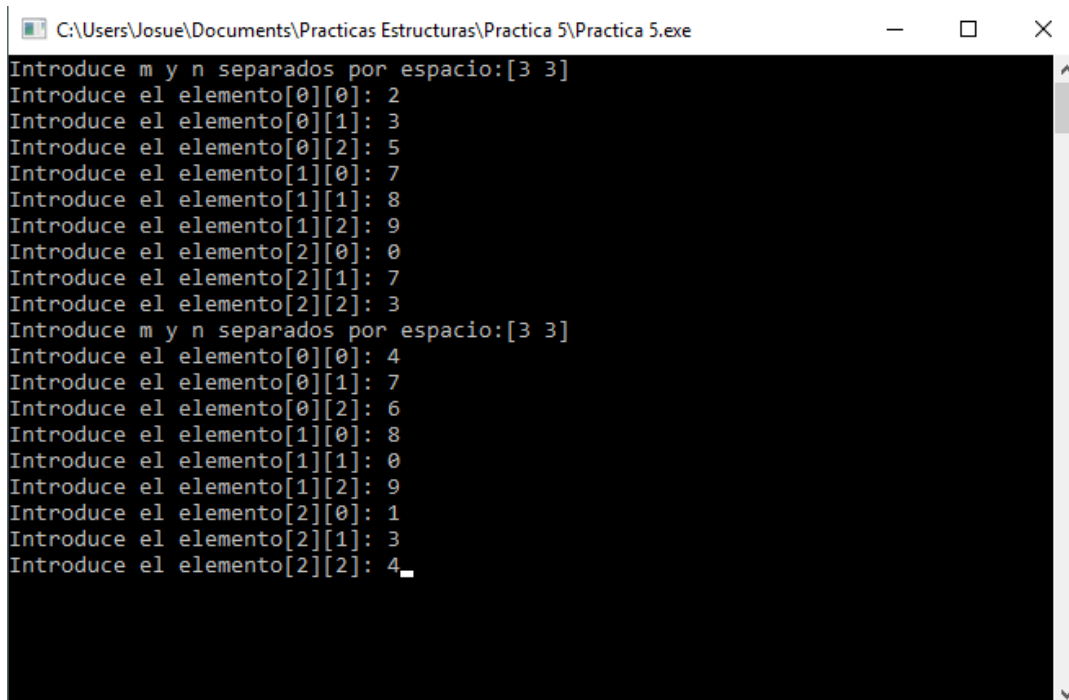


```
C:\Users\Josue\Documents\Practicas Estructuras\Practica 5\Practica 5.exe

*****Matriz Original 1*****
  1  2
  3  4
*****Matriz Original 2*****
  1  2
  3  4
*****Suma de las matrices*****
  2  4
  6  8
*****Resta de las matrices*****
  0  0
  0  0_
```

### Ejemplo matriz 3x3

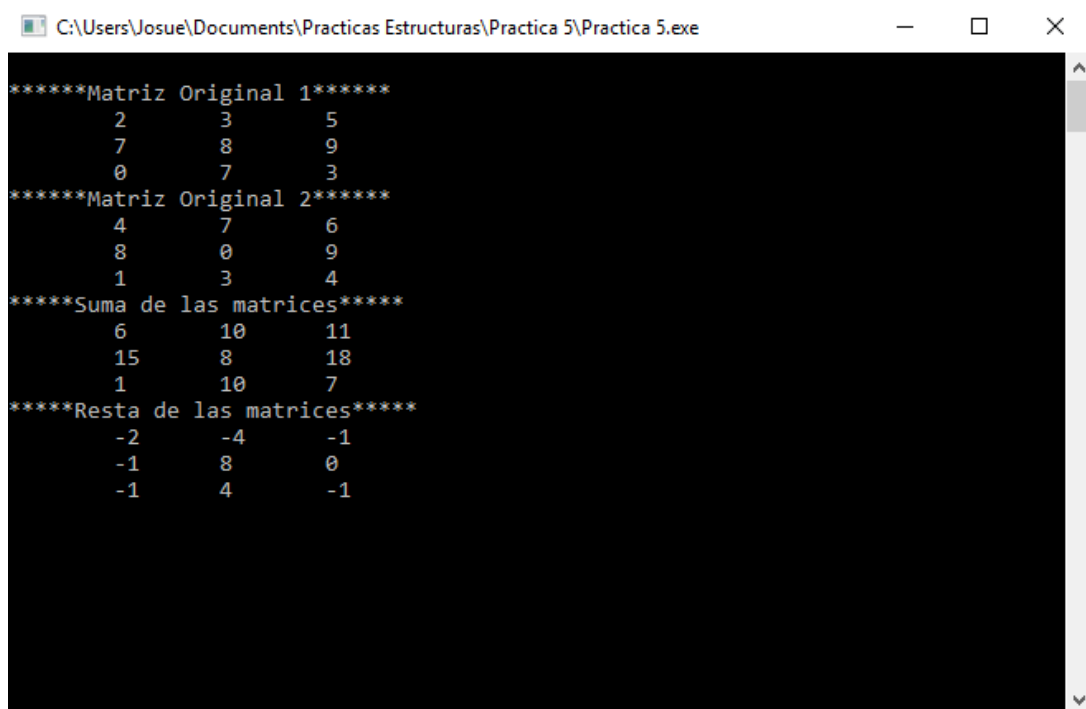
De la misma forma que el ejemplo anterior el programa pide el tamaño y valores que contendrá la matriz para después acomodarlas y aplicar las operaciones de suma y resta



```
C:\Users\Josue\Documents\Practicas Estructuras\Practica 5\Practica 5.exe

Introduce m y n separados por espacio:[3 3]
Introduce el elemento[0][0]: 2
Introduce el elemento[0][1]: 3
Introduce el elemento[0][2]: 5
Introduce el elemento[1][0]: 7
Introduce el elemento[1][1]: 8
Introduce el elemento[1][2]: 9
Introduce el elemento[2][0]: 0
Introduce el elemento[2][1]: 7
Introduce el elemento[2][2]: 3
Introduce m y n separados por espacio:[3 3]
Introduce el elemento[0][0]: 4
Introduce el elemento[0][1]: 7
Introduce el elemento[0][2]: 6
Introduce el elemento[1][0]: 8
Introduce el elemento[1][1]: 0
Introduce el elemento[1][2]: 9
Introduce el elemento[2][0]: 1
Introduce el elemento[2][1]: 3
Introduce el elemento[2][2]: 4_
```

**Después de realizar la limpieza de pantalla muestra los resultados:**



The screenshot shows a Windows command prompt window titled "C:\Users\Josue\Documents\Practicas Estructuras\Practica 5\Practica 5.exe". The window displays the following text:

```
*****Matriz Original 1*****  
    2    3    5  
    7    8    9  
    0    7    3  
*****Matriz Original 2*****  
    4    7    6  
    8    0    9  
    1    3    4  
*****Suma de las matrices*****  
    6   10   11  
   15    8   18  
    1   10    7  
*****Resta de las matrices*****  
   -2   -4   -1  
   -1    8    0  
   -1    4   -1
```