**Instituto Politécnico Nacional**

**ESCOM**

**Práctica 5**

**Arreglos**

**Algoritmia y Programación Estructurada**

**Docente: Mtro. Miguel Ángel Rodríguez Castillo**

**Alumnos: García King Ricardo Neftaly**

**Vaquera Aguilera Ethan Emiliano**

**1CV4**

Índice

Planteamiento del Problema 2

Diseño y Funcionamiento de la Solución 3

Implementación de la Solución 6

Funcionamiento 9

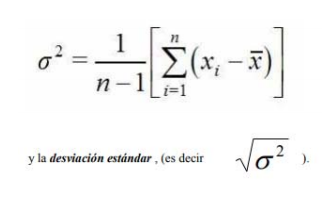
Conclusiones y Bibliografía 15

Anexos 16

Planteamiento del Problema

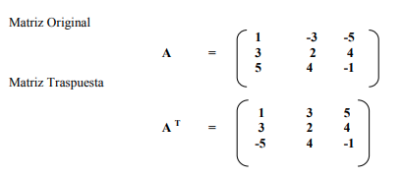
1.-Desarrolla un programa que muestre la multiplicación de los tres números mayores de una serie de números positivos, suponiendo que los datos se leen uno a uno y se van almacenando en un Arreglo con capacidad máxima para 100 números. Considera que un valor de cero como entrada indicará que se ha alcanzado el final de la serie de números positivos.

2. Desarrollar un programa pueda calcular la varianza y desviación estándar de un Arreglo de números reales capturado por el usuario, dada por la fórmula:



3.- Escribir un programa para mostrar la traspuesta de una matriz proporcionada por el usuario.

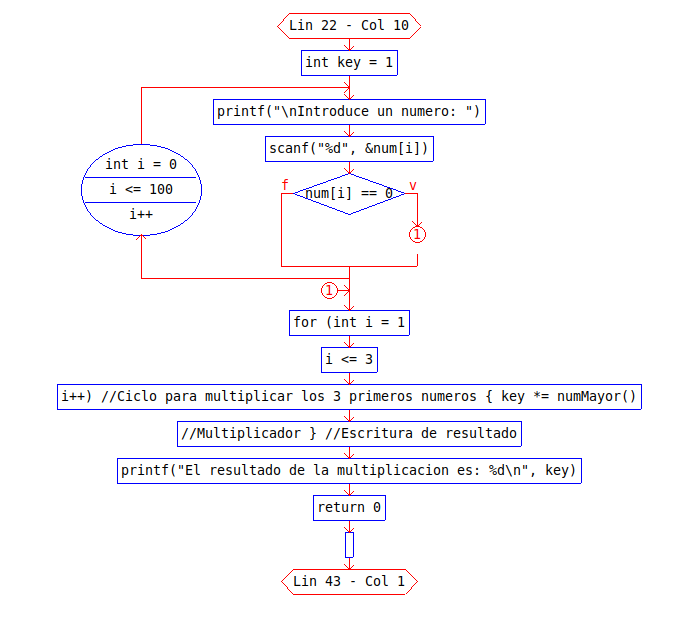
Ejemplo:



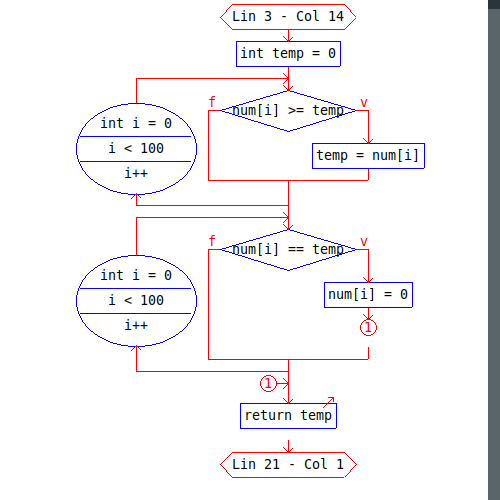
Diseño y funcionamiento de la solución

1.-

**MAIN**

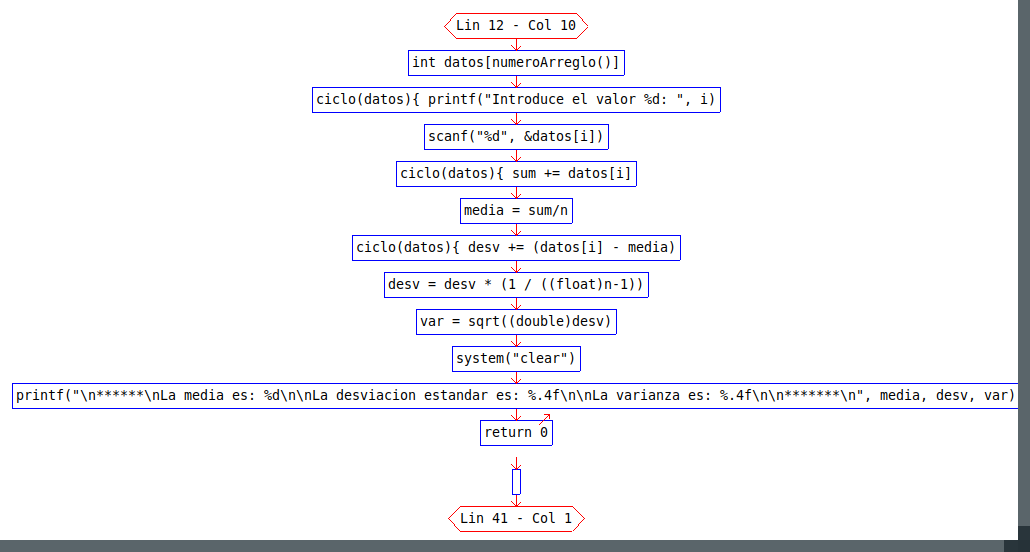


**NumMayor**

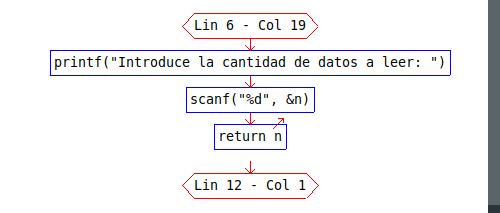


2.-

**MAIN**

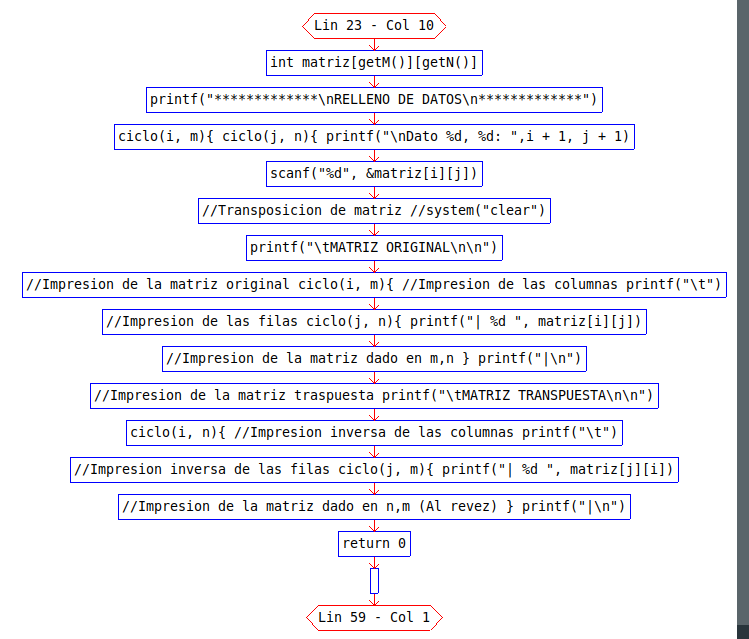
****

**NumeroArreglo**

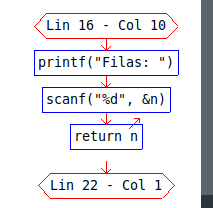
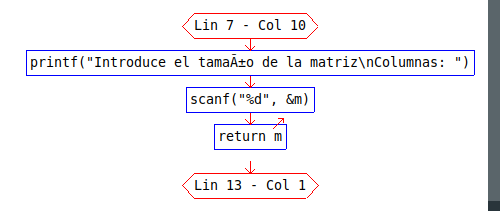
****

3.-

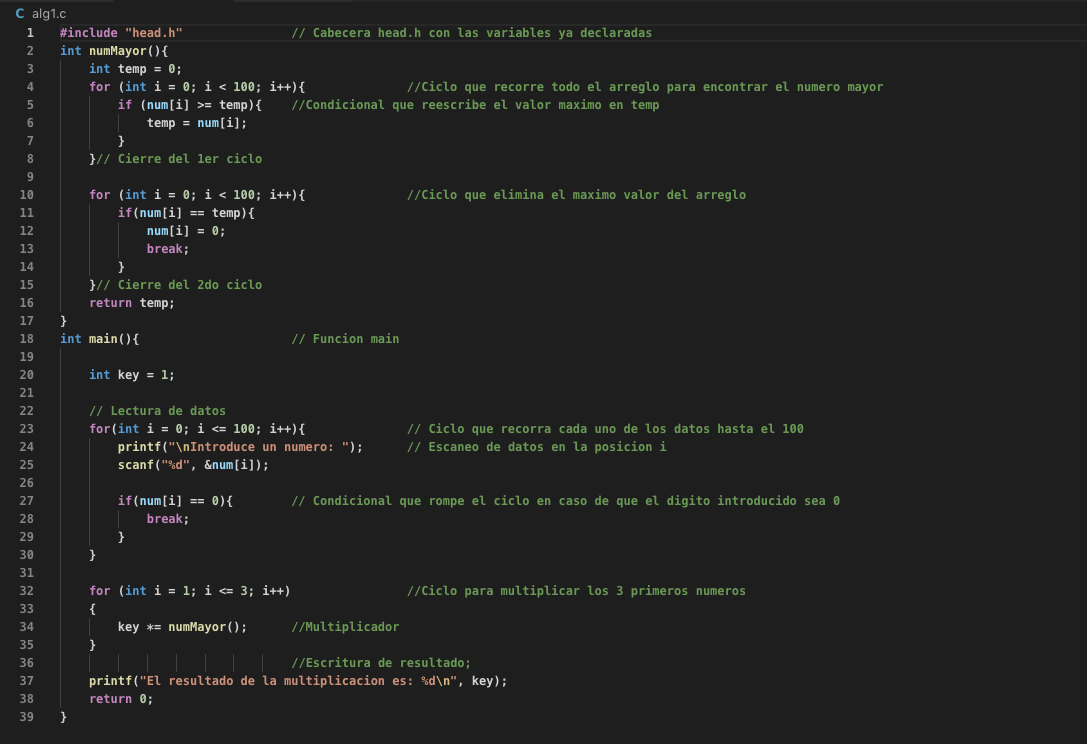
**MAIN**

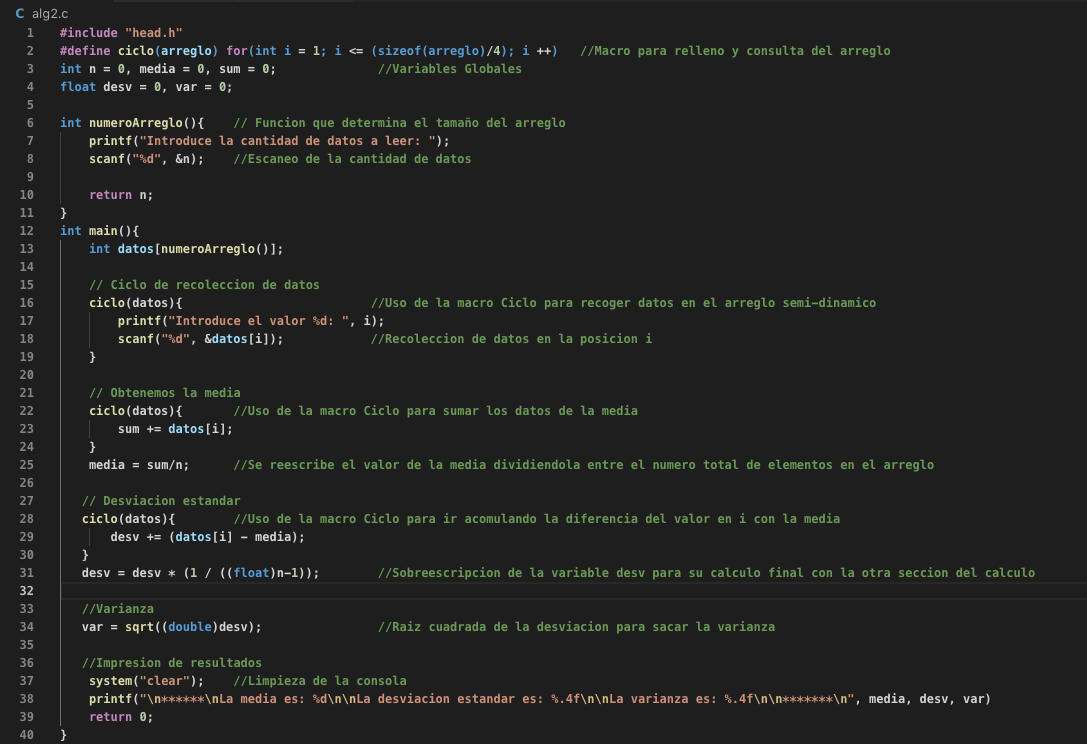
****

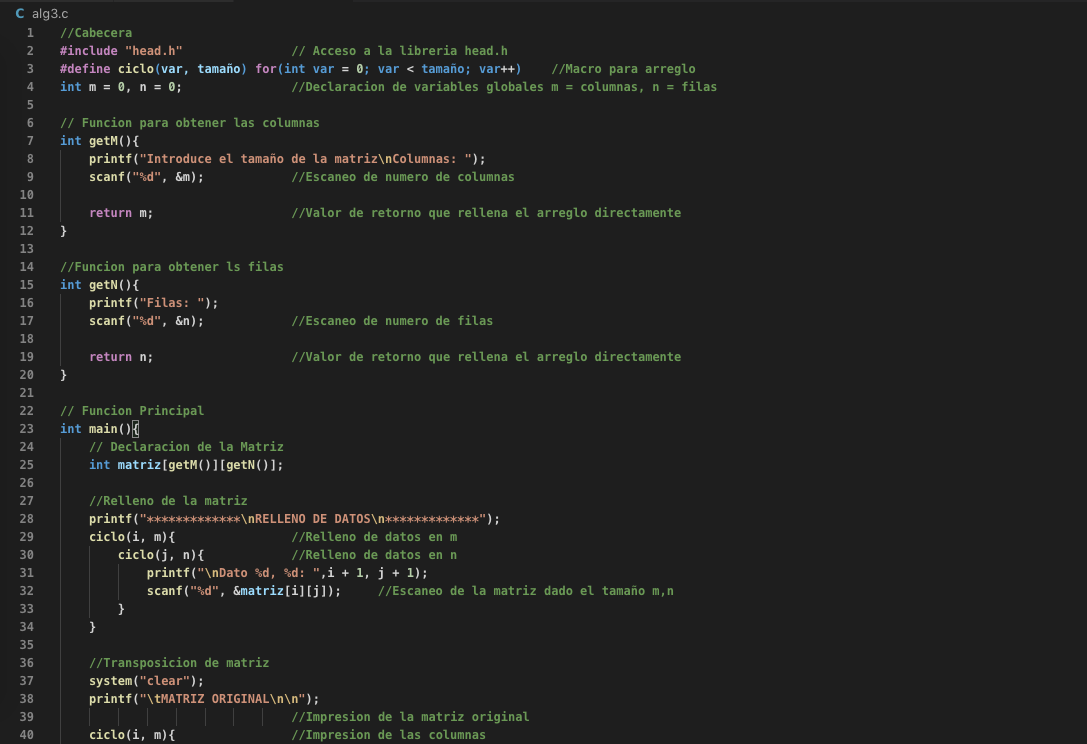
**getM getN**

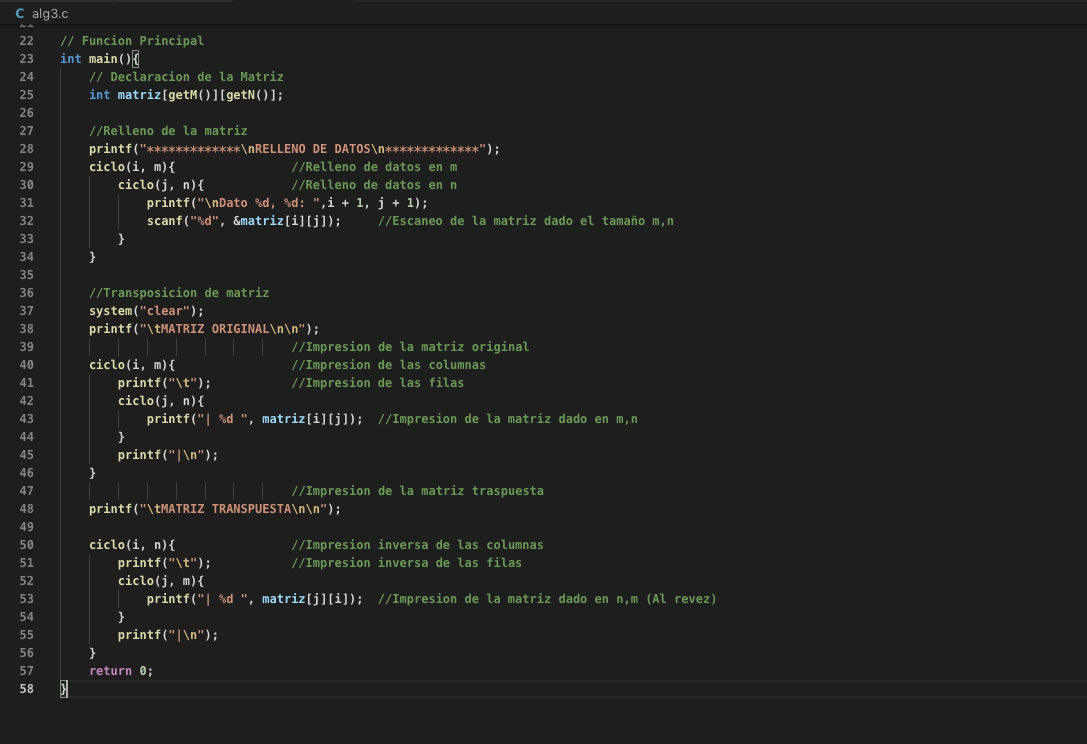
****

Implementación de la Solución

1.-

2.-

3.-



Funcionamiento

Algoritmo 1.-

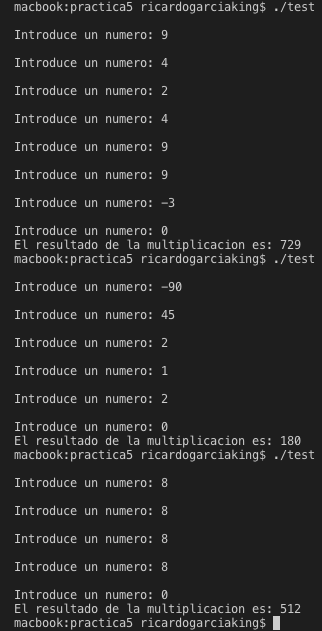
* Prueba de Escritorio

|  |  |
| --- | --- |
| **num** | **resultado** |
| {9, 4, 2, 4, 9, 9, -3, 0} | 729 |
| {-90, 45, 2, 1, 2, 0} | 180 |
| {8, 8, 8, 8, 0} | 512 |

* Resultados de Salida

|  |  |  |
| --- | --- | --- |
| **ENTRADA** | **PROCESO** | **SALIDA** |
| num | int key = 1;  for(int i = 0; i <= 100; i++){  printf("\nIntroduce un numero: ");  scanf("%d", &num[i]);  if(num[i] == 0){  break;  }  }  for (int i = 1; i <= 3; i++){  key \*= numMayor();  } | key |

* Captura de Pantalla



Algoritmo 2.-

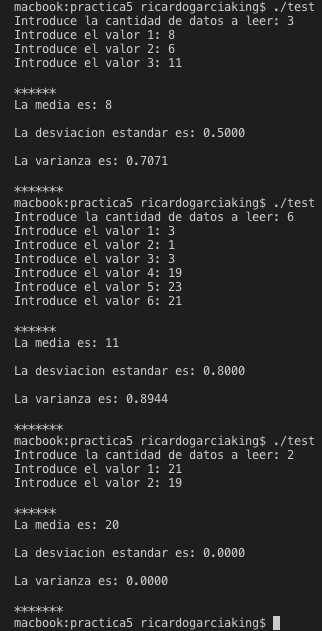
* Prueba de Escritorio

|  |  |  |
| --- | --- | --- |
| **datos** | **varianza** | **desviación estándar** |
| {8, 6, 11} | 0.5000 | 0.7071 |
| {3, 1, 3, 19, 23, 21} | 0.8000 | 0.8944 |
| {21, 19} | 0.0000 | 0.0000 |

* Resultados de Salida

|  |  |  |
| --- | --- | --- |
| **Entrada** | **Proceso** | **Salida** |
| datos[] | ciclo(datos){  printf("Introduce el valor %d: ", i);  scanf("%d", &datos[i]);  }  ciclo(datos){  sum += datos[i];  }  media = sum/n;  ciclo(datos){  var += (datos[i] - media);  }  var = var \* (1 / ((float)n-1));  desv = sqrt((double)var); | var, desv |

* Captura de Pantalla



Algoritmo 3.-

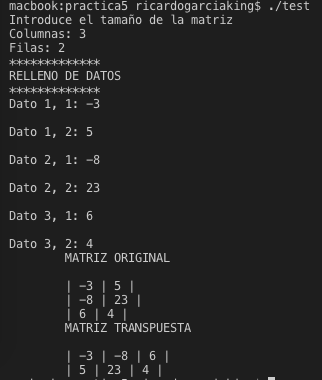
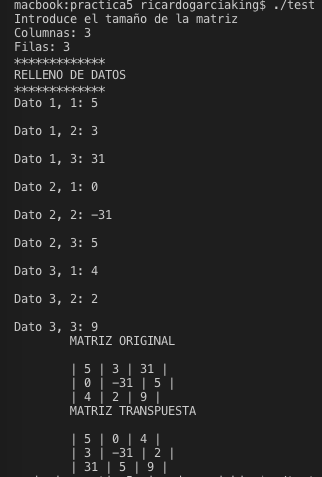
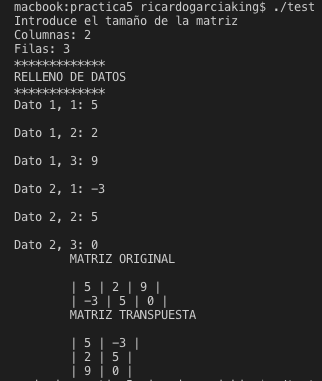
* Prueba de Escritorio

|  |  |
| --- | --- |
| **matriz** | **matriz Traspuesta** |
| { {5, 2, 9},  {-3, 5, 0} } | { { 5, -3},  {2, 5},  {9, 0} } |
| { {5, 3, 31},  {0, -31, 5},  {4, 2, 9} } | { {5, 0, 4},  {3, -31, 2},  {31, 5, 9} } |
| { {-3, 5},  {-8, 23},  {6, 4} } | { {-3, -8, 6},  {5, 23, 4} } |

* Resultado de Salida

|  |  |  |
| --- | --- | --- |
| **Entrada** | **Proceso** | **Salida** |
| matriz | ciclo(i, m){  ciclo(j, n){  printf("\nDato %d, %d: ",i + 1, j + 1);  scanf("%d", &matriz[i][j]);  }  }  ciclo(i, n){  printf("\t");  ciclo(j, m){  printf("| %d ", matriz[j][i]);  }  printf("|\n"); | matriz traspuesta |

* Capturas de Pantalla



Conclusiones y Bibliografía

1.- Se investigó información sobre creación de arreglos dinámicos pero se optó por esperar para la práctica de apuntadores, eso nos llevó a descubrir los macros[[1]](#footnote-0) y a su fundamental implementación en el algoritmo.

Anexos

* Implementación de macros para arreglos “semi-dinámicos”
* En el tercer algoritmo no hubo problemas de dificultad al resolver el problema con arreglos bidimensionales, pero aun así debido a la sintaxis de C, se tuvo que repasar documentación ya que se cometía errores al momento de diferenciar entre filas y columnas.

1. <https://gbenussi.wordpress.com/2013/07/29/macros-en-c-y-c/> [↑](#footnote-ref-0)