Ingeniero en Software y tecnologías emergentes

Materia: Programación Estructurada / Clave **36276**

Alumno: Antonio Ramos González

Matrícula: 372576

Maestro: Pedro Núñez Yépiz

Actividad No. 9: FUNCIONES y METODOS DE ORDENACION Y BUSQUEDA (Anexo)

Tema - Unidad 5 : Arreglos y Cadenas

Ensenada Baja California a 5 de octubre del 2023

RGA_Act9_932

```c
// Mandamos a llamra nuestra libreria
#include "Babilonia.h"
void MatRand(int mtz[][4], int m, int n, int lim_in, int lim_sup);
void PrintMat(int mtz[][4], int m, int n);

// Declaramos nuestras funciones
int msg();
void menu();

int main()
{
    // Genera semilla para numeros aleatoreos
    srand(time(NULL));
    menu();
    return 0;
}

// Menu de eleccion
int msg()
{
    system("CLS");
    printf("MENU\n");
    printf("1.-llenar vector\n");
    printf("2.-llenar matriz\n");
    printf("3.-imprimir vector\n");
    printf("4.-imprimir matriz\n");
    printf("5.-ordenar vector\n");
    printf("6.-buscar en vector\n");
    printf("0.-salir\n");
    printf("Elije una opcion");
    // Retorna el valor de de eleccion
    return valid("Escoje una opcion: ", 0, 6);
}
```

```c
void menu()
{
    // Variables que definen a nuestro vector y matriz
    system("CLS");
    int vect[15];
    int mtz[4][4];
    int opc, bus, num;

    do
    {
        // Recive el valor de nuestro menu
        opc = msg();
        switch (opc) //
        {
        case 1:
            vectRand(vect, 15, 100, 200); // Manda a llenar el vector de con numeros aleatorios
            system("CLS");
            printf("**************************\n");
            printf("El vector ha sido llenado\n");
            printf("**************************\n");
            system("PAUSE");
            break;
        case 2:
            MatRand(mtz, 4, 4, 1, 16); // Manda a llenar la matriz con numeros aleatorios
            printf("**************************\n");
            printf("La matriz ha sido llenada\n");
            printf("**************************\n");
            system("PAUSE");
            break;
        case 3:
            PrintVect(vect, 15); // Imprime vector
            break;
        case 4:
            PrintMat(mtz, 4, 4); // Imprime matriz
            break;
        case 5:
            OrderVect(vect, 15); // Ordena vector de menor a mayor
            system("CLS");
            printf("**************************\n");
            printf("El vector ha sido ordenado\n");
            printf("**************************\n");
            system("PAUSE");
            break;
        case 6:

            system("CLS");
            bus = valid("Que numero deseas buscar: ", 100, 200); // valoda el numero buscado dentro del rango
            num = SearchVect(vect, 15, bus);                     // busca numero en vector
            system("CLS");
            if (num != -1) // Controla si se encuentra el numero fue encontrado
            {
                printf("Numero encontrado en el indice: %d\n", num); // imprime el indice del numero encontrado
            }
            else
            {
                printf("Numero no encontrado\n");
            }
            system("PAUSE");
            break;
```

```c
            if (num != -1) // Controla si se encuentra el numero fue encontrado
            {
                printf("Numero encontrado en el indice: %d\n", num); // imprime el indice del numero encontrado
            }
            else
            {
                printf("Numero no encontrado\n");
            }
            system("PAUSE");
            break;

        default:
            break;
        }
    } while (opc != 0); // Controla que se repita el ciclo
}

// llena la matriz de manera random
void MatRand(int mtz[][4], int m, int n, int lim_in, int lim_sup)
{
    int vect[16];
    int fil, colum, i = 0;
    vectRand(vect, 16, lim_in, lim_sup); // Ob
    system("CLS");
    // variables locales que accederan a los indices de los vectores y la matriz
    for (fil = 0; fil < m; fil++) // ciclo que controla las filas de la matriz
    {

        for (colum = 0; colum < n; colum++) // ciclo que controla las columnas de la matriz
        {

            mtz[fil][colum] = vect[i]; // asigna el valor de vect1 a la matriz
            i++;                       // aumenta contador para acceder al indice del vect1
        }
    }
}

// imprime la matriz
void PrintMat(int mtz[][4], int m, int n)
{
    int fil, colum; // valiables locales
    system("CLS");
    printf("\tMATRIZ 4x4\n");
    for (fil = 0; fil < m; fil++) // ciclo que controla las filas de la matriz
    {

        for (colum = 0; colum < n; colum++) // ciclo que controla las columnas de la matriz
        {
            printf("[%d]\t", mtz[fil][colum]); // imprime matriz
        }
        printf("\n");
    }
    system("PAUSE");
}
```

```
34
35  void menu()
36  {
37      // Variables que definen a nuestro vector y matriz
38      system("CLS");
39      int vect[15];
40      int mtz[4][4];
41      int opc, bus, num;
42
43      do
44      {
45          // Recive el valor de nuestro menu
46          opc = msg();
47          switch (opc) //
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
MENU
1.-llenar vector
2.-llenar matriz
3.-imprimir vector
4.-imprimir matriz
5.-ordenar vector
6.-buscar en vector
0.-salir
Elije una opcionEscoje una opcion: 1
```

```
35  void menu()
36  {
37      // Variables que definen a nuestro vector y matriz
38      system("CLS");
39      int vect[15];
40      int mtz[4][4];
41      int opc, bus, num;
42
43      do
44      {
45          // Recive el valor de nuestro menu
46          opc = msg();
47          switch (opc) //
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
**************************
El vector ha sido llenado
**************************
Presione una tecla para continuar . . .
```

```
37      // Variables que definen a nuestro vector y matriz
38      system("CLS");
39      int vect[15];
40      int mtz[4][4];
41      int opc, bus, num;
42
43      do
44      {
45          // Recive el valor de nuestro menu
46          opc = msg();
47          switch (opc) //
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
MENU
1.-llenar vector
2.-llenar matriz
3.-imprimir vector
4.-imprimir matriz
5.-ordenar vector
6.-buscar en vector
0.-salir
Elije una opcionEscoje una opcion: 2
```

```
37      // Variables que definen a nuestro vector y matriz
38      system("CLS");
39      int vect[15];
40      int mtz[4][4];
41      int opc, bus, num;
42
43      do
44      {
45          // Recive el valor de nuestro menu
46          opc = msg();
47          switch (opc) //
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
**************************
La matriz ha sido llenada
**************************
Presione una tecla para continuar . . .
```

```
35  void menu()
36  {
37      // Variables que definen a nuestro vector y matriz
38      system("CLS");
39      int vect[15];
40      int mtz[4][4];
41      int opc, bus, num;
42
43      do
44      {
45          // Recive el valor de nuestro menu
46          opc = msg();
47          switch (opc) //
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
MENU
1.-llenar vector
2.-llenar matriz
3.-imprimir vector
4.-imprimir matriz
5.-ordenar vector
6.-buscar en vector
0.-salir
Elije una opcionEscoje una opcion: 3
```

```
35  void menu()
36  {
37      // Variables que definen a nuestro vector y matriz
38      system("CLS");
39      int vect[15];
40      int mtz[4][4];
41      int opc, bus, num;
42
43      do
44      {
45          // Recive el valor de nuestro menu
46          opc = msg();
47          switch (opc) //
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
VECTOR
--------------------
[127]
[137]
[143]
[101]
[129]
[147]
[132]
[196]
[146]
[185]
[115]
[138]
[105]
[124]
[151]
Presione una tecla para continuar . . .
```

```
35  void menu()
36  {
37      // Variables que definen a nuestro vector y matriz
38      system("CLS");
39      int vect[15];
40      int mtz[4][4];
41      int opc, bus, num;
42
43      do
44      {
45          // Recive el valor de nuestro menu
46          opc = msg();
47          switch (opc) //
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
MENU
1.-llenar vector
2.-llenar matriz
3.-imprimir vector
4.-imprimir matriz
5.-ordenar vector
6.-buscar en vector
0.-salir
Elije una opcionEscoje una opcion: 4
```

```
35  void menu()
36  {
37      // Variables que definen a nuestro vector y matriz
38      system("CLS");
39      int vect[15];
40      int mtz[4][4];
41      int opc, bus, num;
42
43      do
44      {
45          // Recive el valor de nuestro menu
46          opc = msg();
47          switch (opc) //
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
        MATRIZ 4x4
[6]     [14]    [9]     [5]
[12]    [3]     [7]     [1]
[4]     [13]    [16]    [11]
[15]    [2]     [8]     [10]
Presione una tecla para continuar . . .
```

```
35  void menu()
36  {
37      // Variables que definen a nuestro vector y matriz
38      system("CLS");
39      int vect[15];
40      int mtz[4][4];
41      int opc, bus, num;
42
43      do
44      {
45          // Recive el valor de nuestro menu
46          opc = msg();
47          switch (opc) //
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
MENU
1.-llenar vector
2.-llenar matriz
3.-imprimir vector
4.-imprimir matriz
5.-ordenar vector
6.-buscar en vector
0.-salir
Elije una opcionEscoje una opcion: 5
```

```
34
35  void menu()
36  {
37      // Variables que definen a nuestro vector y matriz
38      system("CLS");
39      int vect[15];
40      int mtz[4][4];
41      int opc, bus, num;
42
43      do
44      {
45          // Recive el valor de nuestro menu
46          opc = msg();
47          switch (opc) //
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
**************************
El vector ha sido ordenado
**************************
Presione una tecla para continuar . . .
```

```
35  void menu()
36  {
37      // Variables que definen a nuestro vector y matriz
38      system("CLS");
39      int vect[15];
40      int mtz[4][4];
41      int opc, bus, num;
42
43      do
44      {
45          // Recive el valor de nuestro menu
46          opc = msg();
47          switch (opc) //
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
VECTOR
----------------------
[101]
[105]
[115]
[124]
[127]
[129]
[132]
[137]
[138]
[143]
[146]
[147]
[151]
[185]
[196]
Presione una tecla para continuar . . .
```

```
35  void menu()
36  {
37      // Variables que definen a nuestro vector y matriz
38      system("CLS");
39      int vect[15];
40      int mtz[4][4];
41      int opc, bus, num;
42
43      do
44      {
45          // Recive el valor de nuestro menu
46          opc = msg();
47          switch (opc) //
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
MENU
1.-llenar vector
2.-llenar matriz
3.-imprimir vector
4.-imprimir matriz
5.-ordenar vector
6.-buscar en vector
0.-salir
Elije una opcionEscoje una opcion: 6
```

```
35  void menu()
36  {
37      // Variables que definen a nuestro vector y matriz
38      system("CLS");
39      int vect[15];
40      int mtz[4][4];
41      int opc, bus, num;
42
43      do
44      {
45          // Recive el valor de nuestro menu
46          opc = msg();
47          switch (opc) //
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
Que numero deseas buscar: 137
```

```
35  void menu()
36  {
37      // Variables que definen a nuestro vector y matriz
38      system("CLS");
39      int vect[15];
40      int mtz[4][4];
41      int opc, bus, num;
42
43      do
44      {
45          // Recive el valor de nuestro menu
46          opc = msg();
47          switch (opc) //
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
Numero encontrado en el indice: 7
Presione una tecla para continuar . . .
```

```
35  void menu()
36  {
37      // Variables que definen a nuestro vector y matriz
38      system("CLS");
39      int vect[15];
40      int mtz[4][4];
41      int opc, bus, num;
42
43      do
44      {
45          // Recive el valor de nuestro menu
46          opc = msg();
47          switch (opc) //
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
Que numero deseas buscar: 131
```

```
35  void menu()
36  {
37      // Variables que definen a nuestro vector y matriz
38      system("CLS");
39      int vect[15];
40      int mtz[4][4];
41      int opc, bus, num;
42
43      do
44      {
45          // Recive el valor de nuestro menu
46          opc = msg();
47          switch (opc) //
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
Numero no encontrado
Presione una tecla para continuar . . .
```

Babilonia.c

```c
// Antonio Ramos Gonzalez Mt: 372576
// 3/10/2023 || 10/5/2023
// Esta biblioteca contiene las funciones que usare durante proximas practicas
// Babilonia
#include<stdio.h>
#include<stdlib.h>
#include<time.h>
#include<string.h>

int valid (const char msg[],int lim_inf,int lim_sup);
void vectRand(int vect[],int m,int lim_inf,int lim_sup);
void PrintVect(int vect[],int m);
void OrderVect(int vect[],int m);
int SearchVect(int vect[],int m, int num);

//Genera numero aleatorios no repetidos en un vector
void vectRand(int vect[],int m, int lim_in, int lim_sup)
{
    int i, x, cont, j, ran;//variables locales
    x = (lim_sup - lim_in) + 1;

    for (i = 0; i < m; i++)//llama a los vectores mediante su indice
    {
        do//ciclo encargado de validar que no se repitan los numeros
        {
            cont = 0;//se encargara de guardar la validacion
            ran = (rand() % x) + lim_in;//genera numero random

            for (j = 0; j < i; j++)//llama a los vectores mediante su indice
            {
                if (vect[j] == ran)//valida que el valor random no se igual a los ya generados
                {
                    cont = 1;
                }
            }

        } while (cont != 0);//repite el ciclo en caso de que se repitan los numeros

        vect[i] = ran;//se asigna el valor random al vecotor
    }
}

//Imprime vector
void PrintVect(int vect[],int m)
{
    int i;//variable local, se encargara de acceder a los indices de los vectores
    system("CLS");
    printf("VECTOR \n");
    printf("----------------------\n");
    for (i = 0; i < m; i++)//ciclo que controla la impresion del vector 1
    {
        printf("[%d]\n",vect[i]);//imprime el vector 1
    }
    system("PAUSE");

}
```

```c

//valida las opciones
int valid(const char msg[], int lim_in, int lim_sup)
{
    char cad[1];//cadena que leera un numero
    int opc;//guardara un numer

    do
    {
        printf("%s", msg);
        fflush(stdin);
        gets(cad);//lee el un numero como caracter
        opc = atoi(cad);//convierte el caracter en un numero

        if (opc < lim_in || opc > lim_sup)//imprime mensaje en caso de que no se valido el numero
        {
            system("CLS");
            printf("SOLO OPCIONES VALIDAS\n");
        }
    } while (opc < lim_in || opc > lim_sup);//valida que el numero sea correcto
    return opc;
}

//ordena un vector
void OrderVect(int vect[],int m)
{
    int i,j;
    int temp;//guarda valor de manera temporal
    for ( i = 0; i < m-1; i++)//Busqueda secuencial
    {
        for ( j = i+1; j < m; j++)
        {
            if(vect[j]<vect[i])
            {
                temp=vect[i];//Guarda el valor de vect[i] en temp
                vect[i]=vect[j];//Guarda el valor de vect[j] en vect[i]
                vect[j]=temp;//Guarda el valor de temp en vect[j]
            }
        }

    }

}

//Busca en el vector
int SearchVect(int vect[],int m,int num)
{
    int i;// define contador
    for ( i = 0; i < m; i++)//
    {
        if(vect[i]==num)//Busca en el vector el numero buscado
        {
            return i; //si encuentra el valor, retorna el valor del indice
        }
    }
    return -1; //si no encuentra el valor, retorna -1
}

```