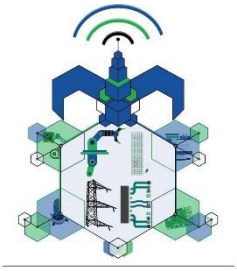


Universidad Veracruzana

Facultad de ingeniería eléctrica y electrónica

Boca del Rio, Veracruz

Experiencia educativa:

Programación estructurada

Docente:

Carlos Arturo Cerón Álvarez

Alumno:

Eduardo Rodriguez Zamora

Actividad:

Ejercicio 1 estructuras

25 de septiembre del 2022

```c
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#define T 50

int menu();
int menu2();
void vector_aleatorio(int *p, int n);
void vector_manual(int *p, int n);
void mostrar(int *p, int n);
void ordenamiento_burbuja(int *p, int n);
void ordenamiento_insercion(int *p, int n);
void burbuja_bidireccional(int *p, int n);
void seleccion(int *p, int n);
void shell_insercion(int *p, int g, int deb);
void shell(int *p, int n);

int main(){
    int opc,opc2;
    int num,*p,i;

    do{
        system("cls");
        opc=menu();
        switch(opc){
            case 1:
                printf("Ingresa el tamanio del vector \n");
                scanf("%d",&num);
                p=(int *)malloc(num*sizeof(int));
                break;
            case 2:
                system("cls");
                opc2=menu2();
                switch(opc2){
                    case 1:
                        vector_aleatorio(p,num);
                        break;
                    case 2:
                        printf("Ingrese valores del vector \n");
                        vector_manual(p,num);
                        break;
                    default: printf("Opcion no valida \n");
                    }
                break;
            case 3:
```

```
                printf("Mostrando contenido del vector \n");
                mostrar(p,num);
                break;
            case 4:
                printf("Ordenamiento de burbuja \n");
                ordenamiento_burbuja(p,num);
                break;
            case 5:
                printf("Ordenamiento de insercion \n");
                ordenamiento_insercion(p,num);
                break;
            case 6:
                printf("Burbuja bidireccional \n");
                burbuja_bidireccional(p,num);
                break;
            case 7:
                printf("Ordenamiento de seleccion \n");
                seleccion(p,num);
                break;
            case 8:
                printf("Ordenacion shell \n");
                shell(p,num);
                break;
            case 0:
                printf("Saliendo.... \n");
                break;
            default: printf("Opcion no valida \n");
        }
        system("PAUSE");
    }while(opc);

    return 0;
}

int menu(){
    int op;

    printf("[1] Crear vector en memoria dinamica \n");
    printf("[2] Llenado del vector \n");
    printf("[3] Mostrar vector \n");
    printf("[4] Ordenacion de burbuja \n");
    printf("[5] Ordenacion por insecion \n");
    printf("[6] Burbuja bidireccional \n");
    printf("[7] Ordenacion por seleccion \n");
    printf("[8] Ordenacion por shell \n");
```

```
    printf("[0] Salir \n");
    printf("Opcion: ");
    scanf("%d",&op);
    return op;

}

int menu2(){
    int op2;
    printf("[1] Llenado aleatorio \n");
    printf("[2] Llenado manual \n");
    printf("Opcion: ");
    scanf("%d",&op2);
    return op2;
}

void vector_aleatorio(int *p, int n){
    int num,i;

    for(i=0;i<n;i++){

      srand(time(NULL));

      num=rand()%T;

    }
    for(i=0;i<n;i++){
    printf("%d\t",*(p+i));
    }
}

void vector_manual(int *p, int n){
    if(n){
        scanf("%d",p);
        vector_manual(++p,--n);
    }
}

void mostrar(int *p, int n){
    if(n){
        printf("%d\t",*p);
        mostrar(++p,--n);
    }
}

void ordenamiento_burbuja(int *p,int n){
```

```
    int i, j;
    int aux;

    for(i=0;i<n;i++){
        for(j=0;j<n;j++){
            if(p[j]>p[j+1]){
                aux=p[j];
                p[j]=p[j+1];
                p[j+1]=aux;
            }
        }
    }
    for(i=0;i<n;i++){
        printf("%d \t", *(p+i));
    }
}

void ordenamiento_insercion(int *p, int n){
    int i,j;
    int aux;

    for(i=0;i<n;i++){
        aux=*(p+i);
        for(j=i;j>0 && p[j-1]>aux;j--){
            p[j]=p[j-1];
        }
        p[j]=aux;
    }
    for(i=0;i<n;i++){
        printf("%d \t", *(p+i));
    }
}

void burbuja_bidireccional(int *p, int n){
    typedef int bool;
    enum{false, true};
    bool permutacion;
    int act=0,dir=1;
    int com=1, fin=n;
    int temp;

    do{
        permutacion=false;
        while(((dir==1) && (act<fin)) || ((dir==-1) && (act>com))){
            act+=dir;
```

```
            if(p[act]<p[act-1]){
                temp=p[act];
                p[act]=p[act-1];
                p[act-1]=temp;
                permutacion=true;
            }
        }
        if(dir==1) fin--; else com++;
        dir=dir-1;
    }while(permutacion);

     for(act=0;act<n;act++){
        printf("%d \t", *(p+act));
    }
}

void seleccion(int *p, int n){
    int act,m_peq,j,aux;

    for(act=0;act<n-1;act++){
        m_peq=act;
        for(j=act;j<n;j++){
            if(p[j]<p[m_peq]){
                m_peq=j;
            }
        }
        aux=p[act];
        p[act]=p[m_peq];
        p[m_peq]=aux;
    }
    for(act=0;act<n;act++){
        printf("%d \t", *(p+act));
    }
}

void shell_insercion(int *p, int n, int deb){
    int j,cours,g;
    int i;
    for(i=g+deb;i<n;i+=g){
        cours=p[i];
        for(j=i;j>=g && p[j-g] > cours;j-=g){
            p[j]=p[j-g];
        }
        p[j]=cours;
    }
```

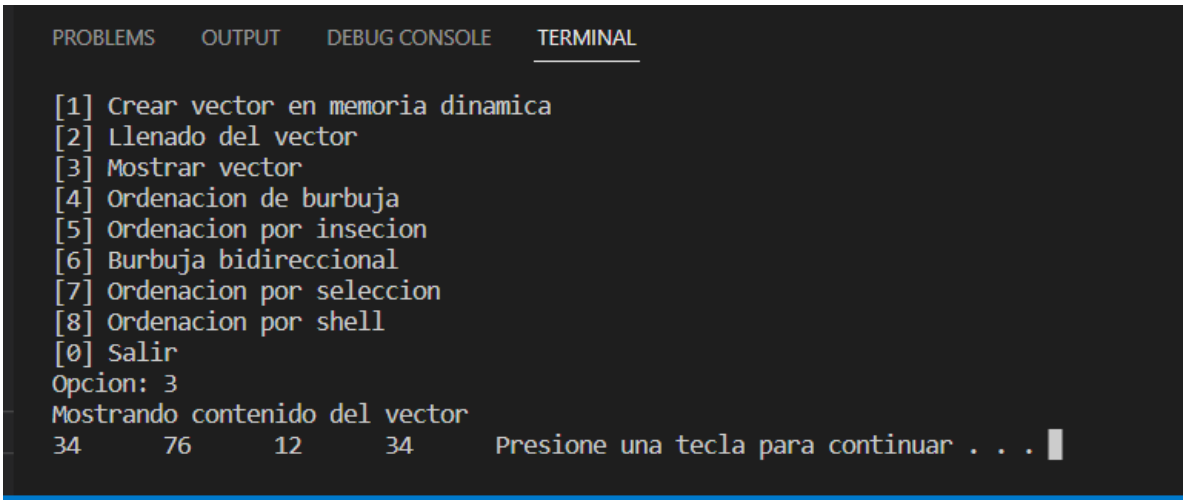
```
}

void shell(int *p, int n){
    int inter[n];
    int gap;
    int i;
    for(gap=0;gap<n;gap++){
        for(i=0;i<inter[gap];i++){
            shell_insercion(p,inter[gap],i);
        }
        for(gap=0;gap<n;gap++){
            printf("%d\t",*(p+gap));
    }
}
}
```

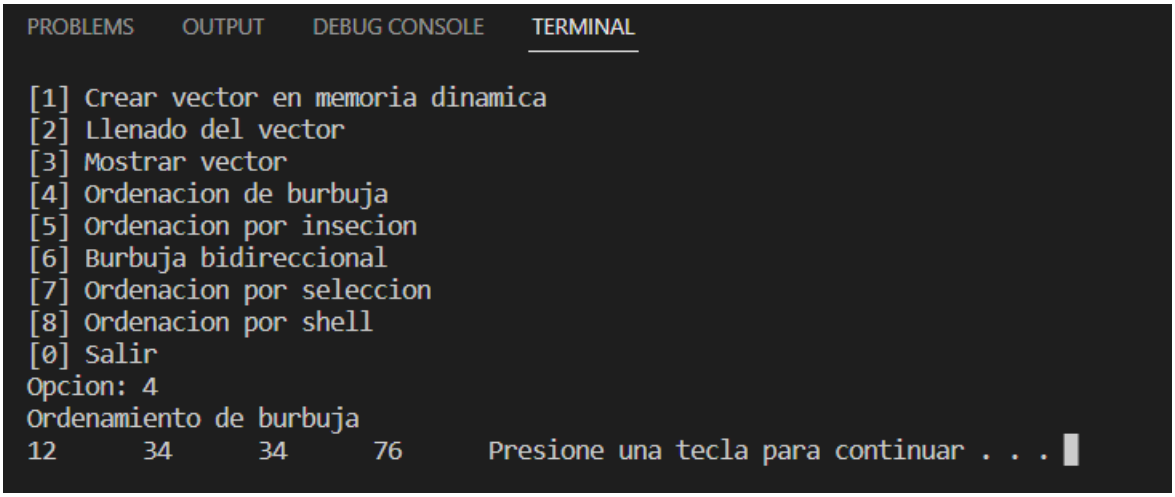
Pruebas de ejecución:

-Ordenación burbuja:

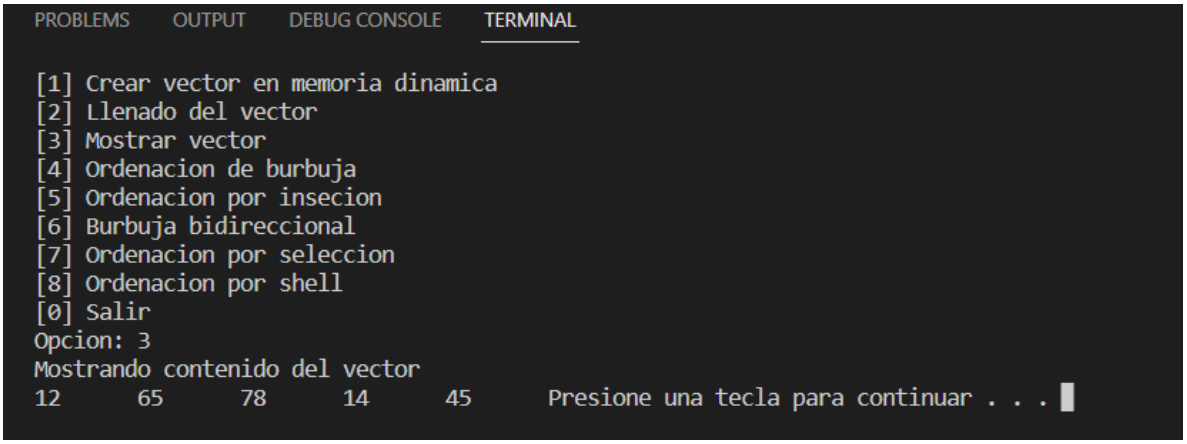
```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL

[1] Crear vector en memoria dinamica
[2] Llenado del vector
[3] Mostrar vector
[4] Ordenacion de burbuja
[5] Ordenacion por insecion
[6] Burbuja bidireccional
[7] Ordenacion por seleccion
[8] Ordenacion por shell
[0] Salir
Opcion: 1
Ingresa el tamanio del vector
4
Presione una tecla para continuar . . .
```

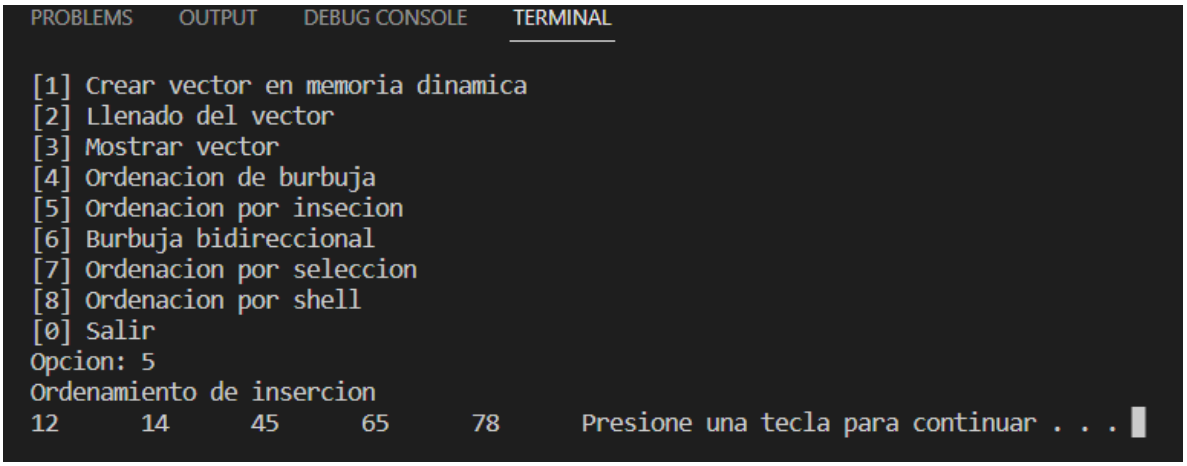
```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL

[1] Llenado aleatorio
[2] Llenado manual
Opcion: 2
Ingrese valores del vector
34
76
12
34
Presione una tecla para continuar . . .
```



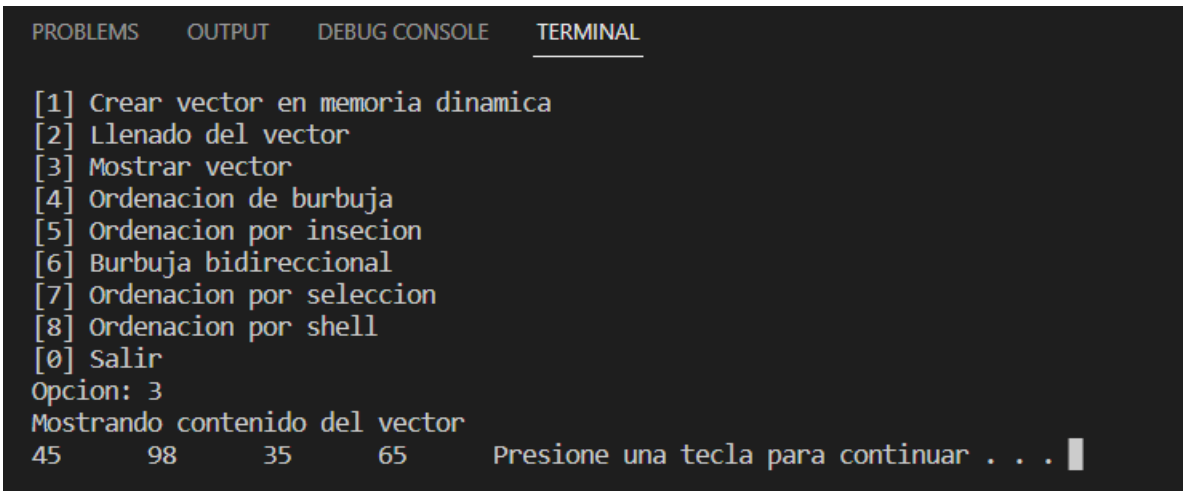


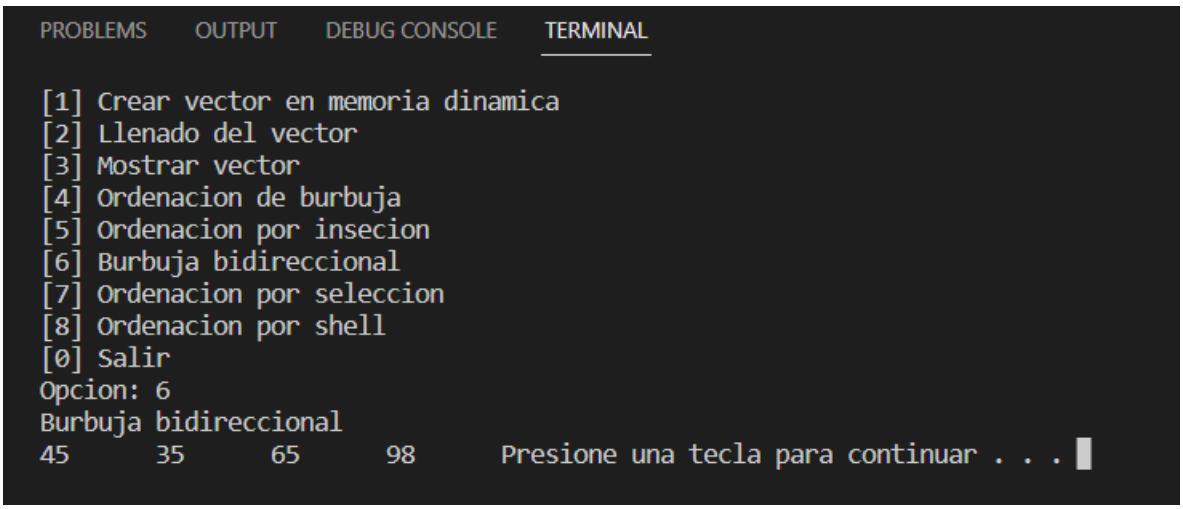
Ordenación por inserción:

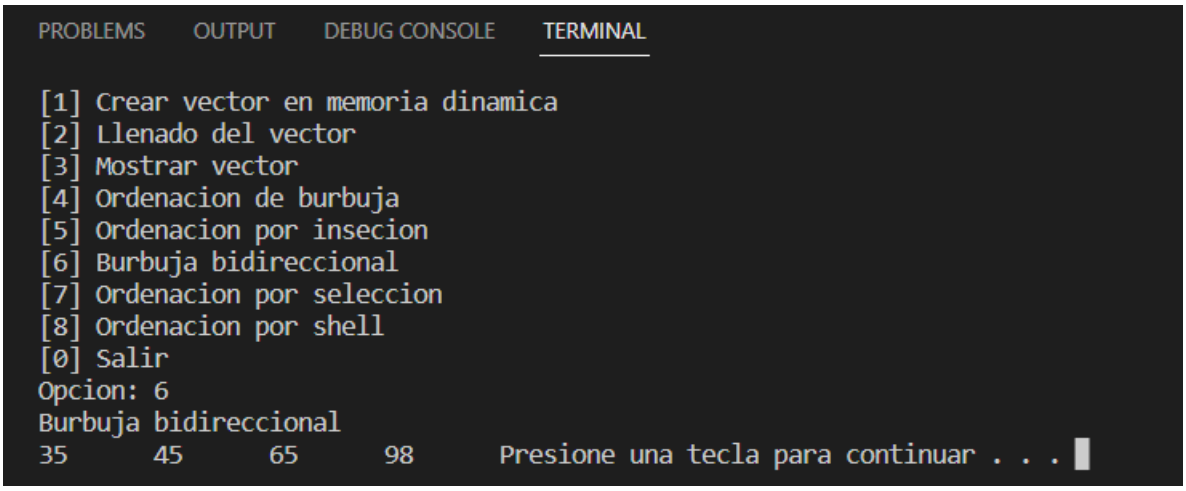




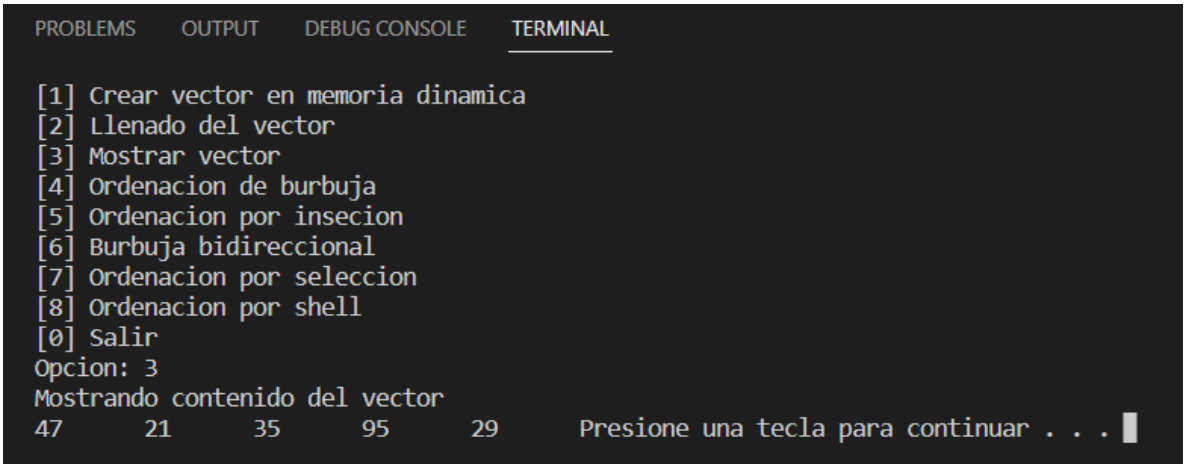
Burbuja bidireccional:

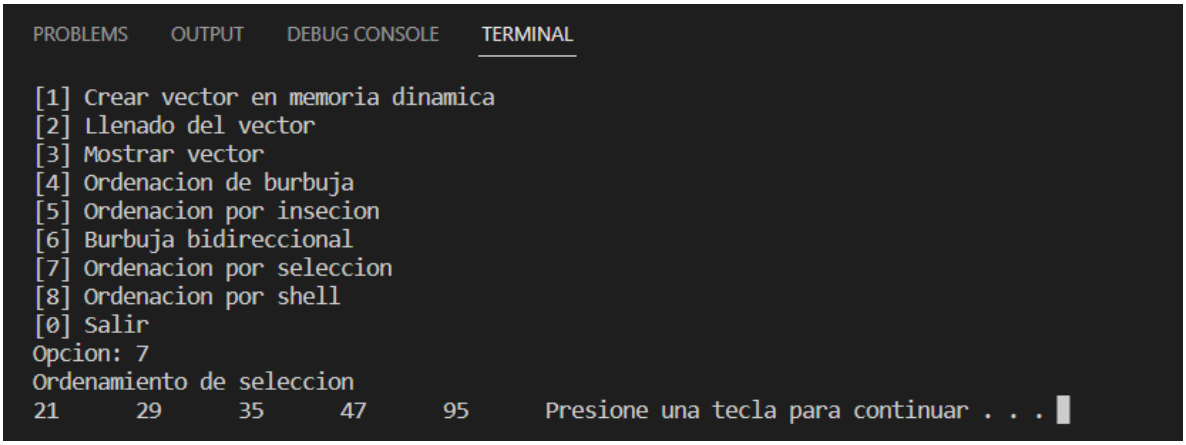




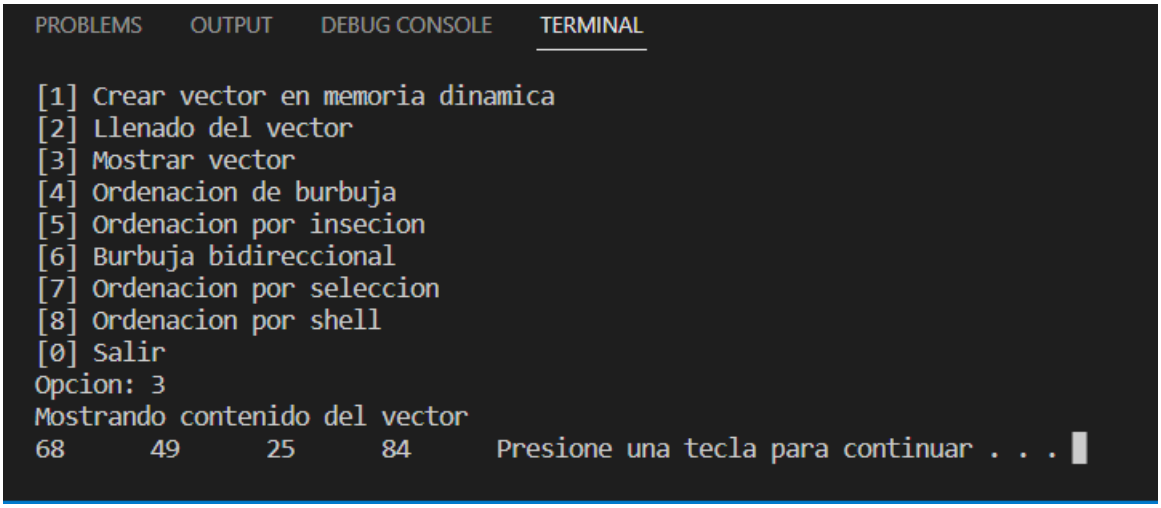


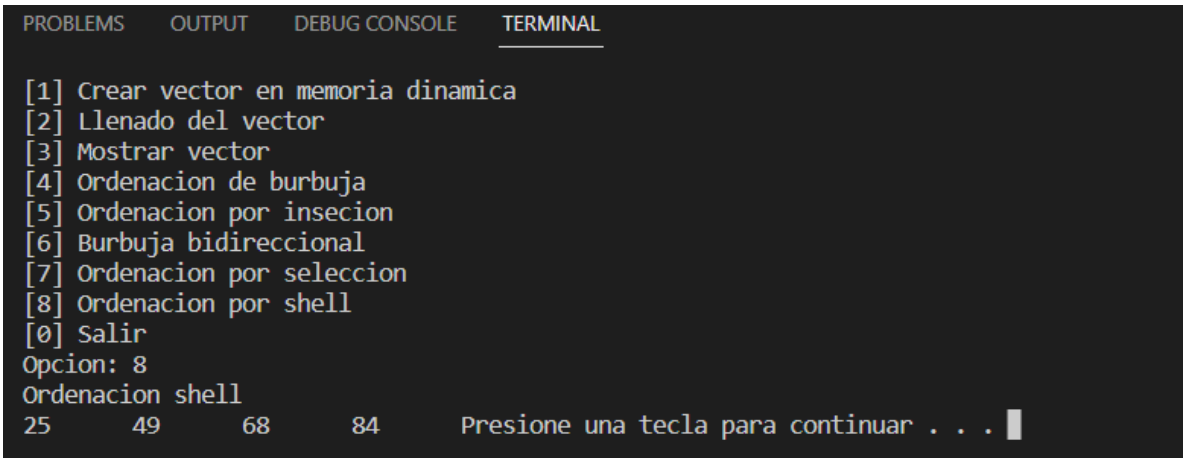
Ordenación por selección:





Ordenación por Shell:





Explicación:

El código esta compuesto por diversas funciones en donde se encuentra el menú que se desplegara al ejecutarse el programa, las funciones que hace ingresar valores al vector de forma manual y aleatoria, una función para mostrar los valores del vector que se ingresa.

Las demás funciones son funciones donde se ejecutan los algoritmos de ordenamiento, que son algoritmos que al tomar el primer numero ingresado este se evaluara con los siguientes comparándolos con una condicional que en caso de que este sea menor se intercambiara la posición con la función apuntador (p[j+1]) o (p[j-1]).