

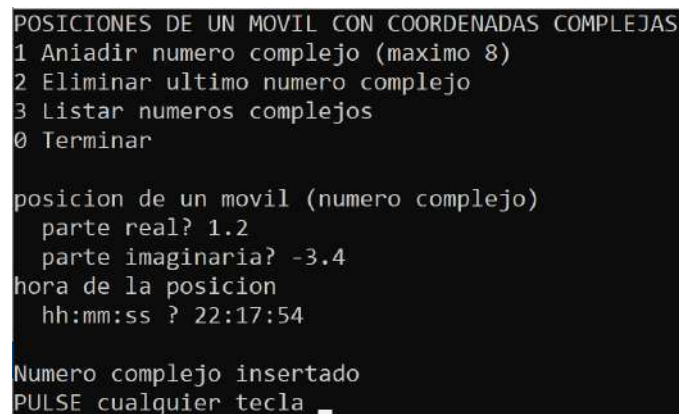
Las sucesivas coordenadas de un móvil se dan como números complejos, que serán almacenadas mediante un programa.

Se usarán las estructuras siguientes:

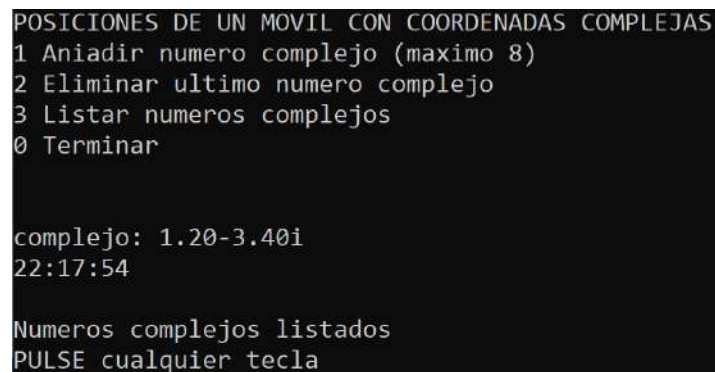
```
struct HORA {  
    unsigned hora,minuto,segundo;  
};
```

```
struct COMPLEJO {  
    float real,imaginaria;  
    struct HORA tiempo;  
};
```

```
struct lista_COMPLEJOS {  
    unsigned total;  
    struct COMPLEJO lista[N];  
};
```



```
POSICIONES DE UN MOVIL CON COORDENADAS COMPLEJAS  
1 Aniadir numero complejo (maximo 8)  
2 Eliminar ultimo numero complejo  
3 Listar numeros complejos  
0 Terminar  
  
posicion de un movil (numero complejo)  
    parte real? 1.2  
    parte imaginaria? -3.4  
hora de la posicion  
    hh:mm:ss ? 22:17:54  
  
Numero complejo insertado  
PULSE cualquier tecla
```



```
POSICIONES DE UN MOVIL CON COORDENADAS COMPLEJAS  
1 Aniadir numero complejo (maximo 8)  
2 Eliminar ultimo numero complejo  
3 Listar numeros complejos  
0 Terminar  
  
complejo: 1.20-3.40i  
22:17:54  
  
Numeros complejos listados  
PULSE cualquier tecla
```

Figura 1. Ejemplos de ejecución del programa

Se definirán, y **se usarán, todas y cada una** de las funciones cuyos prototipos se dan en el siguiente recuadro.

```
// ESPERANZA MACARENA PLAZA MARTINEZ
// Nº de matricula: br0427

#include <stdio.h>
#include <stdlib.h>
#define N 4

//STRUCTS
typedef struct{
    int hora;
    int minuto;
    int segundo;
} HORA;

typedef struct{
    float real;
    float imaginaria;
    HORA tiempo;
} COMPLEJO;

typedef struct{
    int total;
    COMPLEJO arrayComplejo[N];
} ListaComplejos;

//Prototipos de funciones
void printHORA(HORA);
void scanHORA(HORA *);
void scanCOMP (COMPLEJO *);
void printCOMP(COMPLEJO);
void printMenu();
void cualquierTecla();
// espera que se pulse cualquier tecla, sin hacer echo
void inicializarLista(ListaComplejos *);
void addComplejo(ListaComplejos *, COMPLEJO);
void eliminarCOMPLEJO(ListaComplejos *);
// elimina el ultimo complejo añadido a la lista
void printListaCOMPLEJOS(ListaComplejos);
```

```
int main() {
    int seleccion = 0;
    COMPLEJO complejo;
    ListaComplejos lista;

    inicializarLista(&lista);
    do {
        printMenu();
        printf("Escoge opcion: ");
        fflush(stdin);
        scanf("%d", &seleccion);
        switch(seleccion){
            case 1:
                printf("Add numero complejo:\n");
                addComplejo(&lista, complejo);
                printf("Numero complejo insertado\n");
                cualquierTecla();
                break;
            case 2:
                printf("2. Eliminar el ultimo numero\n");
                printf("Numero complejo eliminado\n");
                cualquierTecla();
                break;
            case 3:
                printf("3. Listar numeros complejos\n");
                printListaCOMPLEJOS(lista);
                cualquierTecla();
                break;
            case 0:
                return 0;
            default:
                printf("Opcion incorrecta, vuelva a introducir opcion\n");
        }
    }while(seleccion != 4);

    return 0;
}

//Definicion de funciones
void printHORA(HORA time){
    printf("%.2d:%.2d:%.2d\n", time.hora, time.minuto, time.segundo);
}

void scanHORA(HORA * time){
```

```
    printf("Hora de la posicion (hh:mm:ss): \n");
    fflush(stdin);
    scanf("%d:%d:%d", &time -> hora, &time -> minuto, &time -> segundo);
}

void scanCOMP (COMPLEJO *complejo){
    printf("Parte real: ");
    fflush(stdin);
    scanf("%f", &complejo -> real);
    printf("Parte imaginaria: ");
    fflush(stdin);
    scanf("%f", &complejo -> imaginaria);
    scanHORA(&complejo -> tiempo);
}

void printCOMP(COMPLEJO complejo){
    printf("Posicion de un movil (numero complejo)\n");
    printf("Complejo: %.2f%.2fi\n", complejo.real, complejo.imaginaria);
    printHORA(complejo.tiempo);
}

void printMenu(){
    printf("0. Terminar\n");
    printf("1. Add numero complejo\n");
    printf("2. Eliminar el ultimo numero complejo\n");
    printf("3. Listar numeros complejos\n");
}

void cualquierTecla(){
    printf("PULSE cualquier tecla\n");
    fflush(stdin);
    getchar();
}

void inicializarLista(ListaComplejos * lista){
    lista -> total = 0;
}

void addComplejo(ListaComplejos *lista, COMPLEJO complejo){
    scanCOMP(&(lista -> arrayComplejo[lista -> total]));
    lista -> total++;
}

void eliminarCOMPLEJO(ListaComplejos * lista){
    int pos = 0;
```

```
    if(lista -> total == 0){
        printf("La lista esta vacia");
    } else{
        for(int i = pos; i < lista -> total - 1; i++){
            lista -> arrayComplejo[i] = lista -> arrayComplejo[i + 1];
        }
        lista -> total--;
    }
}

void printListaCOMPLEJOS(ListaComplejos lista){
    for(int i = 0; i < lista.total; i++){
        printCOMP(lista.arrayComplejo[i]);
        printf("\n");
    }
}
```