Práctica I - Arrays, funciones y punteros

1. Escribe un programa **con un menú** que defina las siguientes funciones y emita los resultados:

   a. Cargar un vector de m elementos por teclado. Debes considerar que no podrá superar el máximo de elementos de vector (m=dimensión).

   b. Calcular y emitir la suma de sus elementos.
   c. Calcular y emitir el mínimo del vector.
   d. Calcular y emitir el promedio de los valores del vector
   e. Emitir los valores de aquellos que superaron ese promedio.
   f. Emitir los valores del vector que son múltiplos del último número ingresado en el mismo.
   g. Emitir el valor máximo e indicar la cantidad de veces que apareció y el número de orden en que fue ingresado.
   h. Emitir los valores que son pares.
   i. Emitir los valores que están en posiciones impares.
   j. Emitir aquellos que estén ubicados en posición par.
   k. Recorrer los elementos del vector hasta encontrar el número entero a. Deberá retornar el número de elementos que ha leído hasta encontrar *a*.
   l. Dado un elemento y dado un vector de enteros, realice una función que devuelva el número de apariciones del elemento en el vector. Utiliza el ciclo while para esta función.
   m. Invertir los elementos del vector sin utilizar otro vector. Por ejemplo, el vector formado por los enteros: 1 2 3, debe quedar 3 2 1.

2. Dado un vector de dimensión N. Se debe pedir un valor x (float) y la posición i al usuario para almacenar x en la posición i de un vector. Los valores no se ingresan ordenados por posición. Se debe completar el tamaño del vector. Si la posición está ocupada se vuelve a pedir el ingreso. Al finalizar, emitir el contenido del vector indicando también la posición ocupada por cada número. Utiliza el ciclo while y funciones.

3. Completa e implementa el siguiente programa. Después de cada asignación emite dirección y contenido. Comenta el código:

```
int v[5];
int *p;
...
p = &v[0];
*p = 1;
*(p+1) = 2;
*(p+2) = 3;
```

4. Escribe un programa que imprima los elementos de un vector de enteros en orden inverso utilizando punteros (sin utilizar subíndices [ ]). Pista: obtener la dirección del último elemento y recorrer en orden inverso.

```
int v[10] = {1,2,3,4,5,6,7,8,9,10};

int *p;

……
```

5. Escribe una función que reciba un vector de enteros y su tamaño, y retorne la cantidad de números impares que contiene. Trabaja con notación de punteros y utiliza ciclo while.

```
//cabecera de función:

    int impares (int * v, int tam);
```

6. Escribe una función que retorne un puntero al máximo valor de un vector de “doubles”. Si el vector está vacío debe retornar NULL.

```
//cabecera de función:

    double * max (double * v, int tam);
```

7. Dadas las siguientes declaraciones:

```
    int v[3] = {10,20,30};
```

```
int *p;
p = v;
```

Explica que imprimiría el printf en cada caso de los siguientes:

```
a) (*p)++; printf (“%d”, *p);
b) *(p++); printf (“%d”, *p);
c) *p++; printf (“%d”, *p);
```

8. Qué emite el siguiente programa?:

```
#include <stdio.h>
int main(){
int x[3], *puntero;
x[0]=10;
x[1]=20;
x[2]=30;
puntero = x;
printf("%p\n",puntero);
puntero = &x[0];
printf("%p\n",puntero);
printf("%d\n\n",puntero[0]);
printf("%d\n\n",*puntero);
printf("%X\n\n",&puntero);
printf("%X\n\n",&puntero[1]);
printf("%d\n\n",puntero[1]);
printf("%d\n",*(puntero+1));
printf("%d\n",*(puntero+2));
return 0;
}
```

9. Construye una función tal que dados dos vectores de 5 elementos cada uno, los concatene en un tercer un vector de 10 elementos.

   Ej: V1 = 2-56-7-8-30;
   V2 = 7-80-2-4-13;
   V3 = 2-56-7-8-30-7-80-2-4-13;

10. Ídem anterior, pero los elementos de los dos vectores deben emitirse intercalados. Ej:

    Ej: V4 = 2-7-56-80-7-2-8-4-30-13;

11. Se ingresan los N y M elementos de los arreglos unidimensionales A y B, respectivamente. La computadora emite la unión de ambos, su diferencia y su intersección.

12. Para probar un congelador, la fábrica registra en un listado, la temperatura en el interior durante todos los días del mes de junio. Escribe una función que reciba un vector con todas estas temperaturas (generalmente, negativas) y devuelva la mínima temperatura. Luego escribe una segunda función que diga en qué día del mes se produjo la temperatura mínima.

13. Aritmética de punteros. Investiga el siguiente programa y comenta que emite la última instrucción:

```
int B[] = {3,4,1,2,7,12,-4};
float f = 4.234, *ptf;
*(B+3) = *B + 15;
ptf = &f;
*B = (int)(*ptf);
f = *ptf + 20;
*(B + 5) = (int)(*ptf); // que emite por pantalla B[], f, ptf?
```

14. Qué emite el siguiente programa:

```
#include <stdio.h>
#define SIZE 5
void mystery (int *, int *);
main () {
        inti;
        int x[SIZE] = {2,4,6,8,10};
        int y[SIZE] = {1,3,5,7,9};
        int *xPtr = NULL;
        int *yPtr = NULL;
        mystery (x, y);
            for (i=0; i<SIZE; i++) { printf ("%d\t", x[i]); }
        printf ("\n");
        for (i=0; i<SIZE; i++) { printf ("%d\t", y[i]); }
        printf ("\n");
        return 0;
}
void mystery (int *n1, int *n2)
{
        inti;
        for (i=0; i<SIZE; i++) *(n1+i) = 2 * *(n2+i);
}
```

15. Construye un programa que a partir de dos vectores de enteros con sus valores ordenados crecientemente, cree un tercer vector, también con los datos ordenados en forma creciente. Los dos vectores que se pretenden fusionar no tendrán elementos repetidos en sí mismos, pero entre ellos pueden tener elementos comunes. En este caso, no debe haber repeticiones en el vector que resulte de su fusión.
16. Se dispone de un vector de números enteros de tamaño N, ordenados en forma creciente. Se desea conocer si un número dado introducido desde el teclado se encuentra en la lista. En caso afirmativo, averiguar su posición y emitirla por pantalla. En caso negativo se desea insertarlo en la posición adecuada y posteriormente mostrar la posición por pantalla. Utilizar funciones y un menú.
17. Escribe un programa en el que se defina un vector de *10 punteros a float*, se lean diez números en las ubicaciones en las que hacen referencia cada uno de los punteros del vector de punteros. Cuando este completo, se sumen todos los números y se almacene el resultado en una dirección a la que haga referencia un puntero ajeno al vector. El programa deberá mostrar el contenido de todas las variables, tanto los punteros como los datos de tipo float.
18. Considerando las siguientes declaraciones y sentencias:

```
int array[] = {1,2,3,4,5,6};
int *puntero, x;
puntero = array;
puntero++;
*puntero = *puntero + 6;
puntero = puntero + 3;
puntero = puntero-puntero[-2];
x = puntero - array;
```

a) ¿Cuál es el valor de x?
   a. 1, 2, 3 ó 4?
b) ¿Cual es el valor de array[1]?
   a. 2, 4, 6 ú 8?