

# LENGUAJES DE PROGRAMACIÓN

Trabajo Práctico - Junio de 2020

# Índice

<u>Índice</u>	2
1. <u>Ejercicio 1</u>	3
I. <u>Código</u>	3
II. <u>Explicación</u>	4
III. <u>Pruebas</u>	4
2. <u>Ejercicio 2</u>	7
I. <u>Código</u>	7
II. <u>Explicación</u>	9
III. <u>Pruebas</u>	10
3. <u>Ejercicio 3</u>	13
I. <u>Código</u>	13
II. <u>Explicación</u>	14
III. <u>Pruebas</u>	15
4. <u>Ejercicio 4</u>	15
I. <u>Código</u>	15
II. <u>Explicación</u>	19
III. <u>Pruebas</u>	21

## Ejercicio 1.

### I. Código

```
#include <math.h>
#include <iostream>
using std::cout;
using std::cin;

int main() {
    float n1, n2, o1, omax, o2;
    //1.1
    cout << "\nIntroduce el valor del indice de refracción del
primer medio: ";
    cin >> n1;
    cout << "\nIntroduce el valor del indice de refracción del
segundo medio: ";
    cin >> n2;
    cout << "\nIntroduce el valor del angulo de incidencia: ";
    cin >> o1;
    //1.2
    if(n1 < 1 || n2 < 1 || o1 < 0 || o1 > M_PI / 2){
        cout << "\nLos datos introducidos no son validos";
        return 1;
    }
    //1.3
    omax = asin(n2 / n1);
    if (o1 >= omax)
        cout << "\nNo se produce refracción";
    else {
        o2 = asin(n1 * sin(o1) / n2);
        cout << "El angulo de refraccion es " << o2;
    }
    return 0;
}
```

## II. Explicación

En 1.1 utilizo cout y cin para pedir los valores y almacenarlos en las variables adecuadas respectivamente.

En 1.2 compruebo si no se cumplen alguna de las condiciones de las variables: n1 tiene que ser mayor o igual a 1, al igual que n2, y o1 tiene que pertenecer al intervalo  $[0, \pi/2]$ . Si no cumplen alguna de esas condiciones entra al if, donde indica que los datos no son validos y termina la ejecución con el valor de salida 1.

En 1.3 calculo el valor del ángulo máximo para el que se produce refracción en esos medios y lo comparo con el ángulo introducido o1. Si se produce refracción calcula el ángulo de refracción y lo imprime. Si no se produce lo indica por pantalla.

## III. Pruebas

Hay tres caminos posibles en la ejecución:

- Los datos están mal: voy a probar con valores no aceptados de cada una de las variables.
  - $n1 = 0.5$ ;  $n2 = 2$ ;  $o1 = 0.3$

```
<terminated> (exit value: 1) ejer1.exe [C/C++ Application] E:\Documentos\uni\4.2\mates\LP\ejer1\Debug\ejer1.exe (12/5/20 17:17)
```

```
Introduce el valor del indice de refracción del primer medio: 0.5
```

```
Introduce el valor del indice de refracción del segundo medio: 2
```

```
Introduce el valor del angulo de incidencia: 0.3
```

```
|
Los datos introducidos no son validos
```

- $n1 = 6$ ;  $n2 = -3$ ;  $o1 = 1$

```
<terminated> (exit value: 1) ejer1.exe [C/C++ Application] E:\Documentos\uni\4.2\mates\LP\ejer1\Debug\ejer1.exe (12/5/20 17:18)
```

```
Introduce el valor del indice de refracción del primer medio: 6
```

```
Introduce el valor del indice de refracción del segundo medio: -3
```

```
Introduce el valor del angulo de incidencia: 1
```

```
|
Los datos introducidos no son validos
```

–  $n_1 = 3; n_2 = 4; o_1 = 2$

```
<terminated> (exit value: 1) ejer1.exe [C/C++ Application] E:\Documentos\uni\4.2\mates\LP\ejer1\Debug\ejer1.exe (12/5/20 17:23)

Introduce el valor del indice de refracción del primer medio: 3
Introduce el valor del indice de refracción del segundo medio: 4
Introduce el valor del angulo de incidencia: 2
|
Los datos introducidos no son validos
```

–  $n_1 = 1; n_2 = 1; o_1 = -1$

```
<terminated> (exit value: 1) ejer1.exe [C/C++ Application] E:\Documentos\uni\4.2\mates\LP\ejer1\Debug\ejer1.exe (12/5/20 17:24)

Introduce el valor del indice de refracción del primer medio: 1
Introduce el valor del indice de refracción del segundo medio: 1
Introduce el valor del angulo de incidencia: -1
|
Los datos introducidos no son validos
```

–  $n_1 = a$

```
<terminated> (exit value: 1) ejer1.exe [C/C++ Application] E:\Documentos\uni\4.2\mates\LP\ejer1\Debug\ejer1.exe (12/5/20 17:14)

Introduce el valor del indice de refracción del primer medio: a
Introduce el valor del indice de refracción del segundo medio:
Introduce el valor del angulo de incidencia:
Los datos introducidos no son validos
```

- No se produce refracción:  $n_1$  es mayor que  $n_2$  y  $o_1$  es mayor o igual al ángulo máximo.

–  $n_1 = 3; n_2 = 1; o_1 = 1$

```
<terminated> (exit value: 0) ejer1.exe [C/C++ Application] E:\Documentos\uni\4.2\mates\LP\ejer1\Debug\ejer1.exe (12/5/20 17:29)

Introduce el valor del indice de refracción del primer medio: 3 1 1
|
Introduce el valor del indice de refracción del segundo medio:
Introduce el valor del angulo de incidencia:
No se produce refracción
```

–  $n_1 = 5; n_2 = 2; o_1 = 0.8$

---

<terminated> (exit value: 0) ejer1.exe [C/C++ Application] E:\Documentos\uni\4.2\mates\LP\ejer1\Debug\ejer1.exe (12/5/20 17:29)

```
Introduce el valor del indice de refracción del primer medio: 5 2 0.8
|
Introduce el valor del indice de refracción del segundo medio:
Introduce el valor del angulo de incidencia:
No se produce refracción
```

- Se produce refracción.

–  $n_1 = 3; n_2 = 6; o_1 = 1.2$

---

<terminated> (exit value: 0) ejer1.exe [C/C++ Application] E:\Documentos\uni\4.2\mates\LP\ejer1\Debug\ejer1.exe (13/5/20 0:24)

```
Introduce el valor del indice de refracción del primer medio: 3 6 1.2
|
Introduce el valor del indice de refracción del segundo medio:
Introduce el valor del angulo de incidencia: El angulo de refraccion es 0.484787
```

–  $n_1 = 4; n_2 = 3; o_1 = 0.5$

---

<terminated> (exit value: 0) ejer1.exe [C/C++ Application] E:\Documentos\uni\4.2\mates\LP\ejer1\Debug\ejer1.exe (13/5/20 0:25)

```
Introduce el valor del indice de refracción del primer medio: 4
Introduce el valor del indice de refracción del segundo medio: 3
Introduce el valor del angulo de incidencia: 0.5
El angulo de refraccion es 0.693502
```

## Ejercicio 2.

### I. Código

```
#include <math.h>
#include <fstream>
#include <iostream>
using std::cout;
using std::cin;

//2.1
struct punto {
    int x, y;
    punto *sig;
};
struct lista_puntos {
    punto *primero;
    punto *ultimo;
};

//2.2
lista_puntos* newList() {
    lista_puntos *lista = new (struct lista_puntos);
    lista->primero = NULL;
    lista->ultimo = NULL;
    return lista;
}

//2.3
void insertar(lista_puntos *lista, int new_x, int new_y) {
    punto *t = new (struct punto);
    punto *iter = lista->primero;
    t->x = new_x;
    t->y = new_y;
    t->sig = NULL;
//2.4
    if (lista->primero == NULL) {
        lista->primero = t;
        lista->ultimo = t;
    } else {
//2.5
        while (iter != NULL) {
            if (iter->x == new_x && iter->y == new_y)
                return;
        }
    }
}
```

```

        iter = iter->sig;
    }
    lista->ultimo->sig = t;
    lista->ultimo = t;
}

}

//2.6
void imprimir_puntos(lista_puntos &lista) {
    if(lista->primero == NULL) {
        punto *t;
        for (t = lista.primero; t != lista.ultimo; t = t->sig)
            cout << "\n(" << t->x << ", " << t->y << ")";
        cout << "\n(" << t->x << ", " << t->y << ")";
    }
}

int main(void) {
//2.7
    float a, b, c;
    float x, y, d, dmax = 0;
    lista_puntos *lista = newList();
    std::ifstream f_puntos;

//2.8
    f_puntos.open("puntos.txt", std::ios::in);
    if (!f_puntos) {
        cout << "\nNo se puede abrir el fichero: ";
        return 1;
    }

//2.9
    cout << "\nIntroduce el valor de a: ";
    cin >> a;
    cout << "\nIntroduce el valor de b: ";
    cin >> b;
    cout << "\nIntroduce el valor de c: ";
    cin >> c;
    if (a == b && a == 0) {
        cout << "\nTanto a como b son iguales a 0";
        return 1;
    }

//2.10
    while (!f_puntos.eof()) {
        if (!(f_puntos >> x) || !(f_puntos >> y)) {
            cout << "Error al leer del fichero";
            return 2;
        }
    }
}

```



```

    }
    //2.11
    d = abs(a * x + b * y + c) / sqrt(a * a + b * b);
    if (d > dmax) {
        dmax = d;
        lista = newList();
        insertar(lista, x, y);
    } else if (d == dmax) {
        insertar(lista, x, y);
    }
}
//2.12
cout << "\nLa distancia maxima es: " << dmax << ", dada por los
puntos:";
imprimir_puntos(*lista);
return 0;
}

```

## II. Explicación

Para guardar la información de los puntos con distancia máxima he creado dos structs en 2.1, lista\_puntos es el tipo de la lista donde guardas los puntos y punto es el tipo de los nodos de la lista, que contiene los valores x e y de cada punto y un puntero al siguiente punto de la lista.

En 2.2, la función newList crea una nueva lista y la inicializa vacía.

En 2.3 la función insertar sirve para meter puntos en la lista. Utilizo el puntero t para guardar la información del punto cuyos datos recibo como parámetros e inicializo el puntero al siguiente nodo a NULL ya que si lo inserto va a ser al final de la lista. El puntero iter sirve para comprobar en 2.3 si el punto ya pertenece a la lista, en cuyo caso no lo inserto y salgo de la función.

En 2.4 trato el caso de recibir una lista vacía, inicializando los punteros de la lista primero y ultimo para que apunten al nuevo punto.

Si la lista no está vacía, en 2.5 compruebo que el punto no esté ya en la lista y lo inserto al final.

Para imprimir los puntos de la lista utilizo la función imprimir\_puntos, declarada en 2.6 y que utiliza un iterador para imprimir punto a punto. Antes de imprimir comprueba que la lista no este vacia.

En 2.7 declaro las variables que voy a utilizar: a, b, c para la recta, x, y, d para cada punto que leo del fichero f\_puntos, y dmax (inicializada a 0 para que) y lista para guardar la distancia máxima y los puntos con dicha distancia.

En 2.8 intento abrir el fichero en modo lectura y si no lo consigo lo indico por pantalla y termino con valor de salida 1.

En 2.9 utilizo cout y cin para pedir y guardar los valores (a, b, c) de la recta y compruebo que no se cumpla  $a=b=0$ . En caso de que se cumpla, lo indico por pantalla y termino con valor de salida 2.

En 2.10 leo del fichero, mientras no haya llegado al final, los valores de los puntos. Si hay algún error en la lectura de los puntos lo indico por pantalla y termino con valor de salida 3. Después, en 2.11 calculo el valor de la distancia del punto leído y la comparo con la distancia máxima guardada. Si es mayor vacío la lista e inserto el nuevo punto, si es igual llamo a la función insertar (2.3) y si es menor no hago nada.

Por último, en 2.12 imprimo la distancia máxima y los puntos de la lista con la función imprimir\_puntos (2.6).

### III. Pruebas

Para probar todos los casos, primero voy a fijarme en el fichero de los puntos. Esto me lleva a tener tres casos distintos:

- El fichero no existe. He renombrado el fichero para que, al buscar “puntos.txt”, no lo encuentre.

```
<terminated> (exit value: 1) ejer2.exe [C/C++ Application] E:\Documentos\uni\4.2\mates\LP\ejer2\Debug\ejer2.exe (14/5/20 14:33)
```

```
No se puede abrir el fichero: |
```

- El formato del fichero (dos números por línea) está mal. He metido “asdfghjk” en el fichero para probar la funcionalidad.

```
<terminated> (exit value: 3) ejer2.exe [C/C++ Application] E:\Documentos\uni\4.2\mates\LP\ejer2\Debug\ejer2.exe (14/5/20 14:34)

Introduce el valor de a: 1
Introduce el valor de b: 1
Introduce el valor de c: 1
Error al leer del fichero
```

- El fichero está bien. Utilizo cuatro grupos de puntos:
  - Grupo 1: contiene puntos que no tienen la distancia máxima a ninguna de las rectas que voy a probar.
    - $\{(0, 0), (0.2, 0.8), (0.3, -0.2), (-0.7, -0.5), (-0.1, 0.5)\}$
  - Grupo 2: contiene puntos que tienen la distancia máxima a la recta ‘ $x=0$ ’.
    - $\{(3, 0), (3, -1), (-3, 3)\}$
  - Grupo 3: contiene puntos que tienen la distancia máxima a la recta ‘ $y=0$ ’.
    - $\{(0, 4), (-2, 4), (2, -4)\}$
  - Grupo 4: contiene puntos que tienen la distancia máxima a la recta ‘ $x=y$ ’. El primer punto forma parte del grupo 2 y los otros del grupo 3.
    - $\{(-3, 3), (-2, 4), (2, -4)\}$

El fichero tiene los grupos ordenados de la forma: primero el grupo 1 desordenado, luego todos los puntos desordenados y después lo anterior repetido, seguido del grupo 1 otra vez. Como en este caso la ejecución depende de los datos introducidos, voy a probar tres rectas distintas y el caso  $a=b=0$ , que no genera una recta.

- Recta horizontal ( $x=0$ ):  $a=1, b=0, c=0$

```
<terminated> (exit value: 0) ejer2.exe [C/C++ Application] E:\Documentos\uni\4.2\mates\LP\ejer2\Debug\ejer2.exe (14/5/20 14:45)

Introduce el valor de a: 1
Introduce el valor de b: 0
Introduce el valor de c: 0
|
La distancia maxima es: 3, dada por los puntos:
(3, 0)
(-3, 3)
(3, -1)
```

- Recta vertical ( $y=0$ ):  $a=0$ ,  $b=1$ ,  $c=0$

```
<terminated> (exit value: 0) ejer2.exe [C/C++ Application] E:\Documentos\uni\4.2\mates\LP\ejer2\Debug\ejer2.exe (14/5/20 14:46)

Introduce el valor de a: 0
Introduce el valor de b: 1
Introduce el valor de c: 0
|
La distancia maxima es: 4, dada por los puntos:
(0, 4)
(2, -4)
(-2, 4)
```

- Recta diagonal ( $x=y$ ):  $a=1$ ,  $b=-1$ ,  $c=0$

```
<terminated> (exit value: 0) ejer2.exe [C/C++ Application] E:\Documentos\uni\4.2\mates\LP\ejer2\Debug\ejer2.exe (14/5/20 14:46)

Introduce el valor de a: 1
Introduce el valor de b: -1
Introduce el valor de c: 0
|
La distancia maxima es: 4.24264, dada por los puntos:
(2, -4)
(-3, 3)
(-2, 4)
```

- Error, no es una recta:  $a=0$ ,  $b=0$ ,  $c=2$

```
<terminated> (exit value: 2) ejer2.exe [C/C++ Application] E:\Documentos\uni\4.2\mates\LP\ejer2\Debug\ejer2.exe (14/5/20 14:55)

Introduce el valor de a: 0
Introduce el valor de b: 0
Introduce el valor de c: 2
|
Tanto a como b son iguales a 0
```

## Ejercicio 3.

### I. Código

```
#include <math.h>
#include <fstream>
#include <iostream>
using std::cout;
using std::cin;
#include <iomanip>

int main(void) {
    //3.1
    float fi[11][11], d_x = 0.1, d_t = 0.1, r;
    int i, j, u = 1;
    r = (u * d_t / d_x) * (u * d_t / d_x);
    std::ofstream f_out;
    f_out.open("resultados.txt", std::ios::out);
    //3.2
    for (j = 0; j <= 10; j++) {
        fi[0][j] = 0;
        fi[10][j] = 0;
    }
    //3.3
    for (i = 1; i < 10; i++)
        fi[i][0] = sin(M_PI * (i * d_x));
    //3.4
    for (i = 1; i < 10; i++)
        fi[i][1] = (1 - r) * fi[i][0] + (r / 2) * (fi[i + 1][0] +
fi[i - 1][0]);
    //3.5
    for (j = 2; j <= 10; j++)
        for (i = 1; i < 10; i++)
            fi[i][j] = 2 * (1 - r) * fi[i][j - 1]
                + r * (fi[i + 1][j - 1] + fi[i - 1][j -
1]) - fi[i][j - 2];
    //3.6
    for (i = 0; i < 11; i++) {
        for (j = 0; j < 11; j++) {
            if (j > 0)
                f_out << " ";
            if (fi[i][j] >= 0)
                f_out << "+";
```

```

        f_out << std::fixed << std::setprecision(4) <<
fi[i][j];
    }
    f_out << "\n";
}
f_out.close();
return 0;
}

```

## II. Explicación

En 3.1 están las variables que utiliza el programa. La matriz donde guarda los resultados es  $fi$ . Los datos para obtener el valor de  $r$  los guardo, a pesar de que  $r$  sea constante e igual a uno en todo el ejercicio, para hacer más fácil la posibilidad de cambiar dichos datos. El fichero  $f\_out$  lo abro en modo escritura.

En el bucle for de 3.2 aprovecho para rellenar tanto la primera como la última fila, que vienen dadas por la ecuación  $\phi(0, t) = 0 = \phi(1, t)$ .

En el bucle for de 3.3 calculo el resto de la primera columna siguiendo la ecuación  $\phi(x, 0) = \sin(\pi \cdot x)$ , teniendo en cuenta que  $x = i \cdot \Delta x$ .

En el bucle for de 3.4 calculo el resto de la segunda columna siguiendo la ecuación  $\phi(i, 1) = (1 - r) \cdot \phi(i, 0) + \frac{r}{2} \cdot (\phi(i + 1, 0) + \phi(i - 1, 0))$

Para terminar de calcular los valores de la matriz, uso los bucles for de 3.5 que calculan columna a columna mediante la ecuación dada en el enunciado, cambiando  $j$  por  $j-1$ :

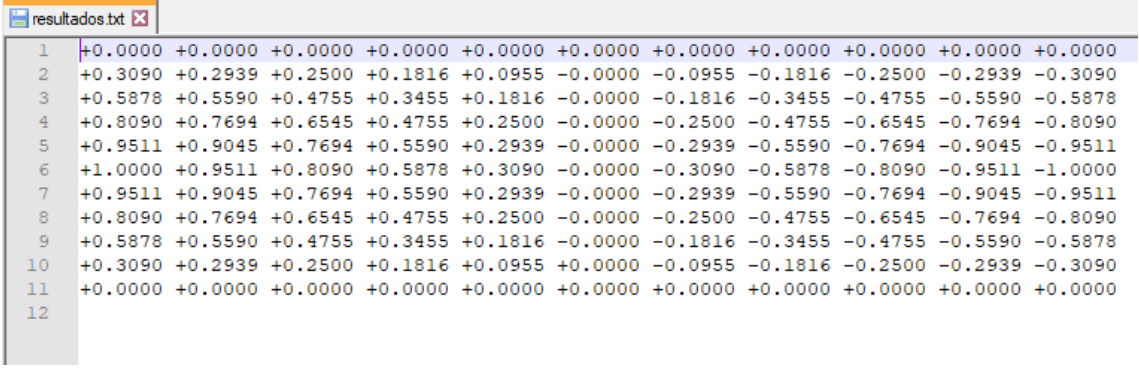
$$\phi(i, j) = (1 - r) \cdot \phi(i, j - 1) + r \cdot (\phi(i + 1, j - 1) + \phi(i - 1, j - 1)) - \phi(i, j - 2)$$

Por último, en los bucles for de 3.6 recorro la matriz fila a fila metiendo los valores en el fichero y con el formato indicado. Utilizo ‘std::fixed’ y ‘std::setprecision(4)’ para obligar a todos los números a tener 4 dígitos de precisión, ‘if (j>0)’ para poner espacios entre las columnas, ‘if (fi[i][j] >= 0)’ para que estén mejor alineados y después de cada for con  $j$  pongo un salto de línea para seguir en la siguiente fila.

### III. Pruebas

El programa no recibe ningún valor por lo que solo habría un resultado posible.

<terminated> (exit value: 0) ejer3.exe [C/C++ Application] E:\Documentos\uni\4.2\mates\LP\ejer3\Debug\ejer3.exe (14/5/20 21:15)



1	+0.0000	+0.0000	+0.0000	+0.0000	+0.0000	+0.0000	+0.0000	+0.0000	+0.0000	+0.0000	+0.0000
2	+0.3090	+0.2939	+0.2500	+0.1816	+0.0955	-0.0000	-0.0955	-0.1816	-0.2500	-0.2939	-0.3090
3	+0.5878	+0.5590	+0.4755	+0.3455	+0.1816	-0.0000	-0.1816	-0.3455	-0.4755	-0.5590	-0.5878
4	+0.8090	+0.7694	+0.6545	+0.4755	+0.2500	-0.0000	-0.2500	-0.4755	-0.6545	-0.7694	-0.8090
5	+0.9511	+0.9045	+0.7694	+0.5590	+0.2939	-0.0000	-0.2939	-0.5590	-0.7694	-0.9045	-0.9511
6	+1.0000	+0.9511	+0.8090	+0.5878	+0.3090	-0.0000	-0.3090	-0.5878	-0.8090	-0.9511	-1.0000
7	+0.9511	+0.9045	+0.7694	+0.5590	+0.2939	-0.0000	-0.2939	-0.5590	-0.7694	-0.9045	-0.9511
8	+0.8090	+0.7694	+0.6545	+0.4755	+0.2500	-0.0000	-0.2500	-0.4755	-0.6545	-0.7694	-0.8090
9	+0.5878	+0.5590	+0.4755	+0.3455	+0.1816	-0.0000	-0.1816	-0.3455	-0.4755	-0.5590	-0.5878
10	+0.3090	+0.2939	+0.2500	+0.1816	+0.0955	+0.0000	-0.0955	-0.1816	-0.2500	-0.2939	-0.3090
11	+0.0000	+0.0000	+0.0000	+0.0000	+0.0000	+0.0000	+0.0000	+0.0000	+0.0000	+0.0000	+0.0000
12											

## Ejercicio 4.

### I. Código

```
#include <string.h>
#include <iostream>
using std::cout;
using std::cin;

//4.1
struct elemento {
    int a;
    int n;
    elemento *sig;
    elemento *ant;
};
typedef struct elemento *polinomio;

//4.2
void insertar(polinomio &poli, elemento *nuevo) {
    elemento *t = poli;
```

```

//4.3
if (poli == NULL) {
    poli = nuevo;
} else if (poli->n > nuevo->n) {
    t->ant = nuevo;
    nuevo->sig = t;
    poli = nuevo;
} else {
    //4.4
    while (t->sig != NULL) {
        if (t->n == nuevo->n) {
            t->a += nuevo->a;
            return;
        } else if (t->sig->n > nuevo->n) {
            t->sig->ant = nuevo;
            nuevo->sig = t->sig;
            t->sig = nuevo;
            nuevo->ant = t;
            return;
        } else
            t = t->sig;
    }
    //4.5
    if (t->n == nuevo->n) {
        t->a += nuevo->a;
        return;
    } else {
        nuevo->ant = t;
        t->sig = nuevo;
    }
}

}

/*
//4.6
void imprimir(polinomio &poli){
    elemento *t = poli;
    cout << "\nimprimir";
    while(t!=NULL) {
        cout << "\n a: " << t->a << ", n: " << t->n;
        t = t->sig;
    }
}
*/

```



```

//4.7
int main(void) {
    std::string str;
    int i, signo;
    float resul, x = 1.5;
    polinomio poli = NULL;
    elemento *aux;
    //4.8
    cout << "\n Introduce el polinomio: ";
    cin >> str;
    //4.9
    i = 0;
    while (i < (int) str.size()) {
        aux = new (struct elemento);
        aux->a = 0;
        aux->n = 0;
        aux->ant = NULL;
        aux->sig = NULL;
        //4.10
        if (str[i] == '+')
            signo = 1;
        else if (str[i] == '-')
            signo = -1;
        else {
            cout << "\nError de formato";
            return 1;
        }
        i++;
        //4.11
        if (str[i] >= 'x')
            aux->a = 1;
        while (str[i] >= '0' && str[i] <= '9') {
            aux->a = 10 * aux->a + str[i] - '0';
            i++;
        }
        aux->a *= signo;
        //4.12
        if (str[i] == 'x') {
            i++;
            if (!(str[i] >= '0' && str[i] <= '9'))
                aux->n = 1;
            while (str[i] >= '0' && str[i] <= '9') {
                aux->n = 10 * aux->n + str[i] - '0';
                i++;
            }
        }
    }
}

```

```

    }
    insertar(poli, aux);
}

//4.13
i = 0;
aux = poli;
while (aux->sig != NULL) {
    if (aux->a != 0) {
        cout << "\na" << aux->n << " = " << aux->a;
        i++;
    }
    aux = aux->sig;
}
if (aux->a != 0) {
    cout << "\na" << aux->n << " = " << aux->a;
    i++;
}
if (i == 0) {
    cout << "\nNo hay ningun coeficiente distinto de 0";
    return 2;
}

//4.14
resul = aux->a;
i = aux->n;
aux = aux->ant;
while (aux->ant != NULL) {
    i--;
    while (i > aux->n) {
        resul = resul * x;
        i--;
    }
    resul = resul * x + aux->a;
    aux = aux->ant;
}
i--;
while (i > aux->n) {
    resul = resul * x;
    i--;
}
resul = resul * x + aux->a;
i--;
while (i > -1) {
    resul = resul * x;
    i--;
}

```

```

    }
    cout << "\np(" << x << ") = " << resul;
    return 0;
}

```

## II. Explicación

4.1 Para guardar el polinomio utilizo una lista (tipo polinomio) ordenada de menor a mayor respecto a  $n$ , donde en cada nodo (tipo elemento) los valores de  $a$  y  $n$  serían  $a_n = a$ . Los nodos tienen un puntero al siguiente elemento de la lista y también al anterior para facilitar la navegación en ambas direcciones en 4.12 y 4.13.

Utilizo enteros para guardar los datos de los coeficientes. El enunciado no indica si los coeficientes van a ser siempre números enteros, pero sí es el caso en el ejemplo. Además, la diferencia sería una transformación del estado (leyendo el coeficiente) a tres estados distintos: (leyendo la parte entera del coeficiente), (leyendo una coma o punto) y (leyendo la parte decimal del coeficiente). En código este cambio se reflejaría en una estructura con dos bucles `while` y una sentencia `if` de forma parecida a la utilizada en 4.11 en vez del bucle `while` de 4.10. Puesto que no aumenta la dificultad del ejercicio, complica la fácil comprensión del código y aumenta el espacio en memoria de los datos, he decidido usar enteros.

4.2 La función `insertar` trata los distintos casos de listas y elementos recibidos:

- 4.3 Cuando el elemento se inserta al principio, ya sea porque la lista está vacía o porque el valor de la  $n$  del nuevo elemento es menor que el del primer elemento de la lista. En el segundo caso adapto los valores de los punteros para mantener la integridad de la lista.
- 4.4 Dentro del bucle `while` compruebo si tengo que insertar el nuevo elemento después del iterador `o`, si es del mismo orden que el nuevo elemento, tengo que sumar los valores de  $a$ .
- 4.5 Después del bucle, si ha llegado, `t` es el último elemento de la lista, por lo que solo hay que comprobar si unirlos porque tienen el mismo  $n$  o insertar al final.

4.6 Una función que imprime la lista que he usado para comprobar fallos.

4.7 Las variables que utilizo son:

- `str`: string donde guardo el polinomio introducido.
- `i`: entero que utilizo para comprobar cada carácter de `str` en 4.8, para comprobar si hay coeficientes distintos a cero en 4.12 y para obtener el resultado correcto en 4.13

- signo: entero que toma los valores 1 o -1 según se lea del substring. Podría usar un booleano y sentencias if else pero me parece más fácil de entender así.
- resul: variable float donde almaceno el resultado de evaluar el polinomio.
- x: variable double que corresponde al valor de x con el que se resuelve el polinomio.
- poli: lista tipo polinomio (4.1) con los datos de cada elemento del polinomio.
- aux: puntero a un elemento que utilizo en 4.8 para guardar los datos que leo del substring e insertarlos en la lista, y en 4.12 y 4.13 para navegar la lista.

4.8 Bucle while que mientras no haya llegado al final del string leído, comprueba el formato de un substring por iteración e inserta los datos en la lista.

4.9 El primer carácter del substring tiene que ser un + o un -, y de este depende la variable signo (1 o -1 respectivamente). Como es el único carácter obligatorio del substring, es el único en el que compruebo si hay errores de formato, y si los hay lo indico y termino. Si no hay ningún error aumento el valor de i para comprobar el siguiente carácter de str.

4.10 Compruebo si después del signo hay o no un número o una x. Si hay una x el valor de a es 1. Si hay un numero (el valor del carácter está entre los valores de '0' y '9' en ASCII) utilizo el bucle para obtener su valor y guardarlo en aux->a. Si no hay ni numero ni x el valor de a se mantiene a 0. Después de la comprobación, multiplico a por signo para obtener el valor con signo.

4.11 Compruebo si el carácter en i es una x. Si es así, aumento el valor de i y compruebo el siguiente carácter. Si el siguiente carácter no es un numero el valor de n es 1. Si hay un numero utilizo el bucle para obtener su valor y guardarlo en aux->n. Si no hay una x al empezar el valor de n se mantiene a 0. Después, al haber comprobado todo el substring, inserto los valores en la lista y compruebo el siguiente substring, si no ha llegado al final del string.

4.12 Utilizo aux para moverme por la lista desde el primer hasta el último elemento. Imprimo los valores de los coeficientes distintos a 0, aumentando el valor de i a la vez. Si al final de la lista i sigue a 0, es decir, no hay ningún coeficiente distinto de 0, lo indico y termino.

4.13 Calculo el resultado de evaluar el polinomio en  $x = 1.5$ , empleando para ello el método descrito en el enunciado del ejercicio. Como aux ha terminado antes apuntando al final de la lista, lo utilizo para navegar por ella en el orden inverso. Para empezar, igualo el resultado al valor de a del ultimo elemento y guardo en i el valor de n. Utilizo i para evaluar los coeficientes iguales a cero que no estén en la lista, ya que no siempre están en el string.

En el bucle avanzo hasta el primer elemento de la lista, evaluando con  $i$  los coeficientes nulos que se encuentran entre los coeficientes de la lista. Después del bucle, compruebo con  $i$  los coeficientes nulos que se encuentren entre el primer y segundo elemento de la lista (si existen), evalúo el coeficiente del primer elemento de la lista y después los coeficientes nulos que existan entre dicho elemento y el coeficiente de grado cero.

Por último, imprimo por pantalla el resultado.

### III. Pruebas

Primero voy a probar el comportamiento de la lista. Defino las funcionalidades de la lista a comprobar:

- F1: insertar al principio de la lista
- F2: insertar en medio de dos nodos
- F3: insertar al final de la lista
- F4: juntar coeficientes del mismo orden

Las cadenas de ejemplo sirven para probar todas las funcionalidades de la lista, y tienen el mismo resultado esperado (-113):

- Prueba 1:  $+1-x+4x^2-16x^5$  (F1 y F3)

---

<terminated> (exit value: 0) ejer4.exe [C/C++ Application] E:\Documentos\uni\4.2\mates\LP\ejer4\Debug\ejer4.exe (18/5/20 14:44)

```
Introduce el polinomio: +1-x+4x2-16x5
|
a0 = 1
a1 = -1
a2 = 4
a5 = -16
p(1,5) = -113
```

- Prueba 2:  $+4x^2+1-16x^5-x$  (F1, F2 y F3)

---

```
<terminated> (exit value: 0) ejer4.exe [C/C++ Application] E:\Documentos\uni\4.2\mates\LP\ejer4\Debug\ejer4.exe (18/5/20 15:36)
```

---

```
Introduce el polinomio: +4x2+1-16x5-x
|
a0 = 1
a1 = -1
a2 = 4
a5 = -16
p(1.5) = -113
```

- Prueba 3:  $-16x^5 - x^2 + 4 + 7x^2 - 3 - x - 2x^2$  (F1, F2, F3 y F4)

---

```
<terminated> (exit value: 0) ejer4.exe [C/C++ Application] E:\Documentos\uni\4.2\mates\LP\ejer4\Debug\ejer4.exe (18/5/20 15:38)
```

---

```
Introduce el polinomio: -16x5-x2+4+7x2-3-x-2x2
|
a0 = 1
a1 = -1
a2 = 4
a5 = -16
p(1.5) = -113
```

Otro aspecto por probar es la correcta impresión de los coeficientes distintos a 0, pero ya se prueba en las anteriores ejecuciones. Queda probar la no impresión de los coeficientes iguales a 0 y el error cuando todos los coeficientes valen 0.

También aprovecho para probar el calculo del resultado. El caso en el que hay coeficientes nulos entre los coeficientes de la lista ya se prueba en las anteriores ejecuciones (1, -1, 4, 0, 0, -16). Quedan por probar los casos con coeficientes nulos entre el primer y segundo coeficiente de la lista y entre el primero y el de grado 0.

- Prueba 4:  $+0x^2 - 2x^5 + 3x^6$  (0, 0, 0, 0, 0, 2, 3) resultado esperado: 18,98

---

```
<terminated> (exit value: 0) ejer4.exe [C/C++ Application] E:\Documentos\uni\4.2\mates\LP\ejer4\Debug\ejer4.exe (18/5/20 15:38)
```

---

```
Introduce el polinomio: +0x2-2x5+3x6
|
a5 = -2
a6 = 3
p(1.5) = 18.9844
```

- Prueba 5:  $-0x+1-2x^4+2x^4-1$

```
<terminated> (exit value: 2) ejer4.exe [C/C++ Application] E:\Documentos\uni\4.2\mates\LP\ejer4\Debug\ejer4.exe (18/5/20 15:41)
```

```
Introduce el polinomio: -0x+1-2x4+2x4-1
|
No hay ningun coeficiente distinto de 0
```

La lectura correcta del formato ya se prueba en las anteriores ejecuciones, solo queda probar los errores de formato.

- Prueba 6:  $+1-x+4x^{216}x^5$  (parecida a la prueba 1 pero quitando un -)

```
<terminated> (exit value: 1) ejer4.exe [C/C++ Application] E:\Documentos\uni\4.2\mates\LP\ejer4\Debug\ejer4.exe (18/5/20 15:42)
```

```
Introduce el polinomio: +1-x+4x216x5
|
Error de formato
```