ESTUDIOS
GENERALES
CIENCIAS

PONTIFICIA
UNIVERSIDAD
CATÓLICA
DEL PERÚ

FUNDAMENTOS DE PROGRAMACIÓN
LABORATORIO 3
PROPUESTAS DE SOLUCIÓN
SEMESTRE ACADÉMICO 2022-2

Horarios: Todos los horarios

Elaborado por Mag. César Aguilera

INDICACIONES:

- Debe utilizar variables descriptivas, comentarios y mensajes descriptivos.
- El orden y la eficiencia de su implementación serán considerados en la calificación.

RESULTADOS ESPERADOS:

- Al finalizar la sesión, el alumno comprenderá el funcionamiento de la estructura algorítmica iterativa con entrada controlada.
- Al finalizar la sesión, el alumno comprenderá el funcionamiento de la iteración controlada por contador.
- Al finalizar la sesión, el alumno construirá programas usando la estructura algorítmica iterativa con entrada controlada.

CONSIDERACIONES:

- La solución presentada para cada problema corresponde a una propuesta de solución por parte del autor.
- En programación pueden existir muchas soluciones para un mismo problema pero debe cumplir con todo lo solicitado, incluyendo las restricciones brindadas.

Desarrolle los siguientes problemas en lenguaje C:

1. Los números enteros romanos - Horarios: 0386, 0387 y 0390

Los números romanos aún son utilizados para algunos propósitos. Los símbolos básicos y sus equivalencias decimales son:

Símbolo	Valor decimal
M	1000
D	500
C	100
L	50
X	10
V	5
I	1

Cuadro 1: Tabla de equivalencias a valores decimales permitidos.

Para escribir los enteros romanos se tienen las siguientes reglas:

- Si una letra está seguida inmediatamente por una de igual o menor valor, su valor se suma al total acumulado. Así, XX = 20, XV = 15 y VI = 6.

- Si una letra está seguida inmediatamente por una de mayor valor, su valor se sustrae del total acumulado. Así, IV = 4, XL = 40 y CM = 900.
- No se pueden ingresar cuatro caracteres del mismo valor. Mostrar ***No se puede ingresar más de tres veces el mismo caracter***.
- Se asume que los caracteres ingresados son siempre los correctos. Ver Cuadro 1

En el Cuadro 1 se pueden apreciar los valores permitidos.

Se pide realizar un programa en lenguaje C, que utilizando estructuras selectivas e iterativas pueda obtener un número decimal a partir del número entero romano ingresado. Se deben de seguir los siguientes pasos:

- Solicite el número de caracteres del número entero romano.
- Luego ingresar cada uno de los caracteres.
- Luego debe mostrar el valor obtenido de la conversión a numero decimal.

Tomado de `http://progra.usm.cl/apunte/ejercicios/2/numeros-romanos.html`

Caso de prueba 1

Ingresar el número de caracteres: 4
Ingresar cada uno de los caracteres:
D
L
I
V
El número decimal (arábigo) es: 554

Caso de prueba 2

Ingresar el número de caracteres: 6
Ingresar cada uno de los caracteres:
M
C
M
X
V
I
El número decimal (arábigo) es: 1916

Caso de prueba 3

Ingresar el número de caracteres: 5
Ingresar cada uno de los caracteres:
C
C
C
I
V
El número decimal (arábigo) es: 304

Caso de prueba 4

Ingresar el número de caracteres: 5
Ingresar cada uno de los caracteres:
C
C
C
C
No se puede ingresar mas de tres veces el mismo caracter.

Programa 1: Propuesta de solución - Números enteros romanos

```
#include<stdio.h>
// los caracteres se ingresan de manera ideal

int main(){
        int i,numcaracter,numArabigo,esError,anterior,valor,valorAnterior,veces, esVeces;
        char letra,letraAnterior;
        printf("=====Ingresar un número entero romano=====\n");
        printf("Indicar el número de caracteres: \n");
        scanf(" %d", &numcaracter);
        printf("Ingresar cada uno de los caracteres: \n");
        i= 1;

        anterior= 0;
        valorAnterior= 0;
        numArabigo= 0;
        veces= 0;
        esVeces= 0;
        while (i<=numcaracter) {
                scanf("\n %c", &letra);
                if (letra=='M')
                        valor=1000;
                else if (letra=='D')
                        valor=500;
                else if (letra=='C')
                        valor=100;
                else if (letra=='L')
                        valor=50;
                else if (letra=='X')
                        valor=10;
                else if (letra=='V')
                        valor=5;
                else if (letra=='I')
                        valor=1;

                if (!esVeces){
                        if (anterior){
                                if(valor<valorAnterior){
                                        numArabigo= numArabigo+valor;
                                        veces= 0;
                                }
                                else if (valor==valorAnterior){
                                        numArabigo= numArabigo+valor;
                                        veces++;
                                }
                                else
                                        numArabigo= numArabigo+valor-2*valorAnterior;
                        }
                        else{
                                numArabigo= numArabigo+valor;
                                anterior=1;
                        }
                        if (veces==3){
```

```
                                esVeces= 1;
                                i=numcaracter;
                        }
                        i++;
                        letraAnterior=letra;
                        valorAnterior=valor;
                }
        }

        if (esVeces)
                printf("No se puede ingresar más de tres veces el mismo caracter");
        else
                printf("El número arábigo es:  %d", numArabigo);
        return 0;
}
```

2. Verificar una fecha - Horarios: B301 y 0391

Las fechas constituyen información muy importante. Podrían ser fechas de interés: la fecha de nacimiento de una persona, la fecha en que empezó a laborar un empleado en una empresa, la fecha de un acontecimiento histórico entre otros.

Un desafío al que se enfrentan frecuentemente las personas es el de comprobar si una fecha es válida. Por ejemplo, se sabe que no todos los meses tienen la misma cantidad de días. Por tal motivo, la fecha "31 de marzo" es válida pero "31 de abril" no. Un caso particular se presenta en el mes de febrero, que normalmente tiene 28 de días, pero si se trata de un año bisiesto, tiene 29.

Un año es bisiesto cuando el número que lo representa es múltiplo de 4 pero no de 100, o cuando es múltiplo de 400.

Para validar si una fecha es correcta se tienen ciertas reglas:

- Según la fecha se debe determinar si el año es bisiesto o no.
- Se debe establecer si la fecha es válida según el año. Verificando el mes y los días del mes.
- Se sabe que los meses de enero, marzo, mayo, julio, agosto, octubre y diciembre tienen 31 días. Además los meses de abril, junio, septiembre y noviembre tienen 30 días.
- Considerar el caso especial del mes de febrero.
- La semana inicia el lunes. A nivel internacional, el estándar ISO 8601 del año 2004 estableció que la semana comienza en lunes y termina en domingo.

Se pide realizar un programa en lenguaje C, que utilizando estructuras selectivas e iterativas al ingresar una fecha se pueda obtener el número del día y el número de la semana de esa fecha en el año. Se deben de seguir los siguientes pasos:

- Solicite el ingreso de una fecha. En formato día, mes y año.
- Ingresar el día que inicia el año. Siendo [D]Domingo, [L]Lunes, [M]Martes, [X]Miércoles, [J]Jueves, [V]Viernes y [S]Sábado .
- Luego debe mostrar los valores obtenidos para el número del día en el año y la semana del año.
- En caso de error debe mostrar mensaje de ***fecha incorrecta***.

Caso de prueba 1

Ingresar fecha [AAAAMMDD]: 20221015
Día de la semana que inicia el año: S
Día del año: 294
Semana del año: 42

Caso de prueba 2

Ingresar fecha [AAAAMMDD]: 20220103
Día de la semana que inicia el año: S
Día del año: 3
Semana del año: 1

Caso de prueba 3

Ingresar fecha [AAAAMMDD]: 20220102
Día de la semana que inicia el año: S
Día del año: 2
Semana del año: 52

Caso de prueba 4

Ingresar fecha [AAAAMMDD]: 20210513
Día de la semana que inicia el año: V
Día del año: 137
Semana del año: 21

Caso de prueba 5

Ingresar fecha [AAAAMMDD]: 20191313
Día de la semana que inicia el año: X
Fecha incorrecta

Programa 2: Propuesta de solución - Verificar una fecha

```
#include<stdio.h>
#define SEMANA 7

int main(){
        int fecha,dia,mes,year,i,total,semana,primeraSemana;
        int pconforme,pdia,pbisiesto;
        char diaSemana;

        printf("Ingresar fecha[aaaammdd]:");
        scanf(" %d",&fecha);
        printf("Día de la semana que inicia el año:");
        scanf("\n %c",&diaSemana);

        year=fecha/10000;
        mes=(fecha %10000)/100;
        dia=(fecha %10000) %100;

        printf(" %d  %d  %d  %d \n",fecha, year,mes,dia);
        pdia=0;
        if (diaSemana=='D'||diaSemana=='L'||diaSemana=='M'||diaSemana=='X'||
```

```
        diaSemana=='J'||diaSemana=='V'||diaSemana=='S')
        pdia=1;
    printf("pdia %d \n",pdia);
    //año bisiesto
    if ((year %4==0 && year %100!=0) || (year %400==0))
        pbisiesto=1;
    else
        pbisiesto=0;

    pconforme=0;
    //valida fecha completa
    printf("pbisiesto %d \n",pbisiesto);
    if (dia<1 || mes<1 || mes>12 || year< 0)
        pconforme=0;
    else{
        if (((mes==1 ||mes==3||mes==5||mes==7||mes==8||mes==10 ||mes==12)&& dia<=31) ||
            ((mes==4 ||mes==6||mes==9||mes==11) && dia<=30))
                pconforme=1;
        else if (mes==2 && !pbisiesto && dia<=28)
                pconforme=1;
        else if (mes==2 && pbisiesto && dia<=29)
                pconforme=1;
    }
    printf("pConforme %d \n",pconforme);
    if (pconforme && pdia){
        i=1;
        total=0;
        while (i<mes){
            if(mes==1||mes==3||mes==5||mes==7||mes==8||mes==10||mes==12)
                total=total+31;
            else if (mes==4|| mes==6 ||mes==9||mes==11)
                total=total+30;
            else
                if(pbisiesto)
                    total=total+29;
                else
                    total=total+28;
        i++;
        }
        total= total+dia;
        if (diaSemana=='S')
            primeraSemana=1;
        else if(diaSemana=='V')
            primeraSemana=2;
        else if(diaSemana=='J')
            primeraSemana=3;
        else if(diaSemana=='X')
            primeraSemana=4;
        else if(diaSemana=='M')
            primeraSemana=5;
        else if(diaSemana=='L')
            primeraSemana=6;
        else
            primeraSemana=7;
        if (total<=primeraSemana)
            semana= 1;
        else{
            total= total−primeraSemana;
            semana= (total/7)+2;
            total= total+primeraSemana;
        }
        printf("Día del año: %d \n", total);
        printf("Semana del año: %d \n", semana);
    }
    else{
        printf("Fecha incorrecta");
    }
```

```
88          return 0;
89  }
```

3. Convertir un número decimal(fraccionario) real a Base N - Horarios: 0380, 0381, 0384 y 0385

Dado de número real, se desea cambiar este número de base DECIMAL a otra base N (el algoritmo no contempla cambio de dígitos a letras en base hexadecimal) que el mismo ingresará como dato.

Para esto es mejor es separar el número en: parte entera y parte fraccionaria. Por ello, primero se transforma un número entero de base decimal a otra base, mediante el siguiente algoritmo:

1. Dividimos el número (entero) entre N.
2. Dividimos el cociente obtenido entre N y repetimos el mismo procedimiento hasta que el cociente sea cero.
3. El número en base N lo formamos tomando el primer dígito del último cociente, seguidos por los residuos obtenidos en cada división, seleccionándolos de derecha a izquierda.

Por ejemplo, como pasar el número 65 a base 4. Veamos la Figura 1

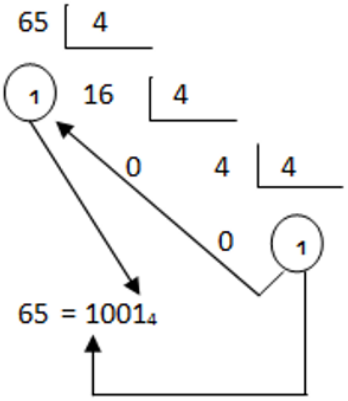

Figura 1: Transformar número decimal a base 4

Con la parte fraccionaria se realiza lo siguiente:

1. Teniendo la cantidad de dígitos de la fracción, es un dato de entrada.
2. Iterar de a uno según la cantidad de dígitos.
   a) Calculamos el producto de la fracción por la base N.
   b) Obtenemos la parte entera del producto.
   c) Comenzamos a formar el número en base N, con el primer dígito de la parte entera. En donde el primer dígito en base N corresponde a la parte entera de la primera iteración, el segundo dígito a la parte entera de la segunda iteración, y así sucesivamente hasta llegar al último.
   d) Al producto le quitamos la parte entera.

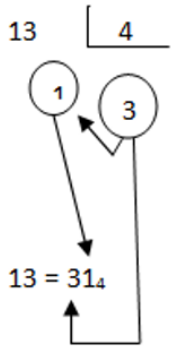

Figura 2: Transformar número parte entera a base 4

Por ejemplo, como pasar 13.3125 a base 4. Con la parte entera se sigue el método anterior. Ver Figura 2

Del ejemplo anterior ahora la fraccionaria. Con 0.3125, tiene 4 dígitos en la parte de la fracción. Ver Figura 3

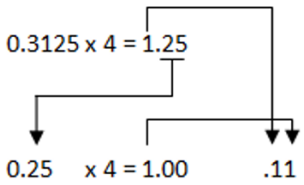

Figura 3: Transformar número parte entera a base 4

Se pide realizar un programa en lenguaje C, que utilizando estructuras selectivas e iterativas al ingresar un número decimal real (fraccionario) y la base N de conversión pueda mostrar el número en base N. Se deben de seguir los siguientes pasos:

- Solicite el ingreso de un número decimal. Debe ser positivo mayor o igual a 1.
- Ingresar la base a convertir. Validar rango de valores entre 2 y 9.
- Ingresar la cantidad de dígitos decimales del número decimal real. Debe ser entero y mayor o igual a cero. Se asume que se ingresa de manera correcta.
- En caso de error debe mostrar mensaje de ***Número inválido*** o ***Número de base fuera del rango*** o ***La cantidad de dígitos decimales no puede ser negativo***.

Caso de prueba 1

Ingresar número decimal fraccionario: 13.3125
Ingresar la base a convertir: 4
Ingresar la cantidad de dígitos decimales del número fraccionario: 4
La parte entera en base 4 es: 31
La parte fraccionaria en base 4 es: 1100
El número obtenido es: 31.11

Caso de prueba 2

```
Ingresar número decimal fraccionario: 65
Ingresar la base a convertir: 4
Ingresar la cantidad de dígitos decimales del número fraccionario: 0
La parte entera en base 4 es: 1001
La parte fraccionaria en base 4 es: 0
El número obtenido es: 1001
```

Caso de prueba 3

```
Ingresar número decimal fraccionario: 32
Ingresar la base a convertir: 2
Ingresar la cantidad de dígitos decimales del número fraccionario: 0
La parte entera en base 4 es: 100000
La parte fraccionaria en base 4 es: 0
El número obtenido es: 100000
```

Caso de prueba 4

```
Ingresar número decimal fraccionario: 25.25
Ingresar la base a convertir: 4
Ingresar la cantidad de dígitos decimales del número fraccionario: 2
La parte entera en base 4 es: 121
La parte fraccionaria en base 4 es: 10
El número obtenido es: 121.10
```

Caso de prueba 5

```
Ingresar número decimal fraccionario: 7
Ingresar la base a convertir: 3
Ingresar la cantidad de dígitos decimales del número fraccionario: 0
La parte entera en base 4 es: 21
La parte fraccionaria en base 4 es: 0
El número obtenido es: 21
```

Caso de prueba 6

```
Ingresar número decimal fraccionario: -121
Ingresar la base a convertir: 5
Ingresar la cantidad de dígitos decimales del número fraccionario: 0
El número es inválido.
```

Caso de prueba 7

```
Ingresar número decimal fraccionario: 124
Ingresar la base a convertir: 5
Ingresar la cantidad de dígitos decimales del número fraccionario: -2
La cantidad de dígitos decimales no puede ser negativa.
```

Programa 3: Propuesta de solución - Convertir un número decimal(fraccionario) real a Base N

```c
#include<stdio.h>
#include<math.h>

int main(){
	int aux,base,cont,digito,invertido,cantdigitos;
	double num;
	printf("Ingresar un número:\n");
	scanf(" %lf",&num);
	printf("Ingresar la base: \n");
	scanf(" %d",&base);
	printf("Ingresar la cantidad de decimales: \n");
	scanf(" %d",&cantdigitos);

	if (num<=0)
		printf("Número inválido\n");
	else if (base<2 || base >9)
		printf("Número de base fuera del rango");
	else if (cantdigitos<0)
		printf("La cantidad de dígitos decimales no puede ser negativo");
	else {
		//cambiar de base
		aux=num;
		cont=0;
		invertido=0;
		while(aux!=0){
			digito=aux %base;
			invertido=digito*pow(10,cont)+invertido;
			aux=aux/base;
			cont++;
		}
		printf("El número en base %d es: %d \n", base, invertido);
		//
		int ent,t_ent,basef,i;
		double dec, numero;
		i=1;
		ent=(int)num;
		dec=(num-ent);
		//printf("parte decimal: %lf \n",dec);
		basef=0;
		while (i<=cantdigitos){
			dec=dec*base;
			t_ent=(int)dec;
			basef=basef*10+t_ent;
			dec=dec-t_ent;
			i++;
		}
		printf("La parte decimal en base %d es %d \n", base, basef);
		if (cantdigitos>0){
			numero=invertido+((double)basef/pow(10,cantdigitos));
			printf("El número obtenido en base %d es %lf", base, numero);
		}
		else
			printf("El número obtenido en base %d es %d", base, invertido);
	}
	return 0;
}
```

4. El método de la posición falsa o regla falsa - Horarios: 0382, 0383, 0388 y 0389

El método de la posición falsa o regla falsa se utiliza para encontrar raíces en ecuaciones de una variable. Este método combina el método de bisección y el método de la secante. Se tiene un rango [a,b] donde f(a) y f(b) tienen signos opuestos, por lo tanto existe un valor r entre a y b en el cual f(r)=0, r viene a ser una raíz de la ecuación.

En la figura 4, se muestra la ecuación f y las secantes trazadas; inicialmente se traza la secante entre a_0 y b_0, siendo b_1 el punto que intersecta el eje X, entonces b_1 reduce el intervalo inicial a $[a_0, b_1]$; luego se traza la secante entre a_0 y b_1, siendo a_2 el punto que intersecta el eje X, entonces se reduce el intervalo a $[a_2, b_1]$; y se continúan realizando estos pasos hasta que el punto que intersecta la secante y el eje X es la raíz de la ecuación.

El punto de la secante que intersecta el eje X, se calcula con la siguiente fórmula: c = b - f(b) * $\frac{(a-b)}{f(a)-f(b)}$

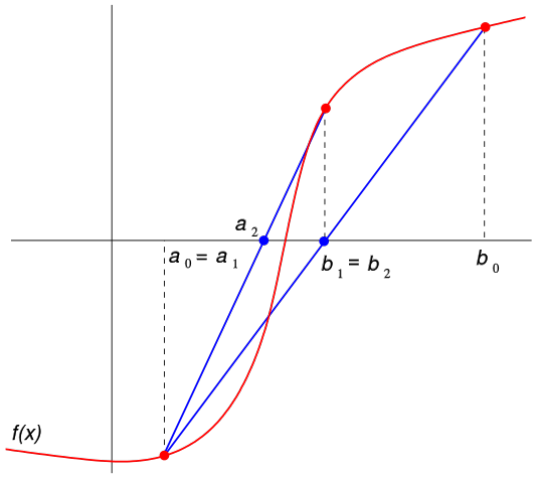

Figura 4: Método de la posición falsa

El método realiza los siguiente:

1. Calcular el punto c (intersección de la secante y el eje X) del intervalo y evaluar la ecuación para ese punto c, f(c).
2. Si f(c) y f(a) tienen signos opuestos, entonces el nuevo intervalo donde evaluar la raíz será [a,c]. Si no es así, entonces el intervalo donde evaluar la raíz será [c,b].
3. Se continúa desarrollando los pasos anteriores hasta que se encuentre la raíz de la ecuación en el punto c.

Se pide realizar un programa en lenguaje C, dados los coeficientes de una ecuación de grado 3, la cantidad de iteraciones, el intervalo a y b y un valor para el error máximo. Calcule la raíz de la ecuación siguiendo el método de la posición falsa. Debe mostrar, en cada iteración, como cambian los intervalos, la evaluación de la función y el error calculado.

También debe validar que el error máximo ingresado sea menor o igual a 0.1, que en el intervalo ingresado a sea menor que b y que f(a) y f(b) tengan signos opuestos.

El error se calcula como la diferencia en valor absoluto entre la raíz real de la función y la raíz aproximada por el método.

Utilice la siguiente ecuación de grado 3: $f(x) = 2x^3 + x - 1$ con raíz real: 0.59

Caso de prueba 1

Ingresar cantidad de iteraciones: 5
Ingresar el máximo valor: 0.001
Ingrese el intervalo [a,b]: 1 0
Los valores del intervalo no son válidos.

Caso de prueba 2

Ingresar cantidad de iteraciones: 4
Ingresar el máximo valor: 0.2
Ingrese el intervalo [a,b]: 0 3
Los valores del del error no es válido.

Caso de prueba 3

Ingresar cantidad de iteraciones: 7
Ingresar el máximo valor: 0.001
Ingrese el intervalo [a,b]: 0 0.5
f(a) y f(b) no tiene signos diferentes, por lo tanto no se puede hallar una raíz para la ecuación en el intervalo ingresado.

Caso de prueba 4

Ingresar cantidad de iteraciones: 9
Ingresar el máximo valor: 0.001
Ingrese el intervalo [a,b]: 0 1

Itera.	a	b	f(a)	f(b)	c	f(c)	error
1	0	1	-1	2	0.33333	-0.59259	0.25667
2	0.33333	1	-0.59259	2	0.48571	-0.28510	0.10429
3	0.48571	1	-0.28510	2	0.54988	-0.11758	0.04012
4	0.54988	1	-0.11758	2	0.57487	-0.04515	0.01513
5	0.57487	1	-0.04515	2	0.58426	-0.01685	0.00574
6	0.58426	1	-0.01685	2	0.58773	-0.00622	0.00227
7	0.58773	1	-0.00622	2	0.58901	-0.00228	0.00099
8	0.58901	1	-0.00228	2	0.58948	-0.00083	0.00052
9	0.58948	1	-0.00083	2	0.58965	-0.00030	0.00035

La raíz calculada es 0.58965, con un error de 0.00053 y cumple con tener un error máximo de 0.001

Programa 4: Propuesta de solución - El método de la posición falsa o regla falsa

```
#include<stdio.h>
#include<math.h>

int main(){
        int i,n,veces,raiz_encontrada;
        double a,b, x3,x2,x1,x0,error;
        printf("Ingrese los coeficientes [x3 x2 x1 x0]: \n");
        scanf(" %lf  %lf  %lf  %lf",&x3,&x2,&x1,&x0);
        printf("Ingrese la cantidad de iteraciones: \n");
        scanf(" %d",&veces);
        printf("Ingrese el error máximo: \n");
        scanf(" %lf",&error);
        printf("Ingrese el intervalo[a b]: \n");
        scanf(" %lf  %lf",&a,&b);
        double fa,fb,fc,c,error_calculado,raiz_calc;

        if (a<b) {
                if (error<= 0.1){
                        i= 1;
                        fa= x0+x1*a+x2*pow(a,2)+x3*pow(a,3);
                        fb= x0+x1*b+x2*pow(b,2)+x3*pow(b,3);
                        if (fa*fb<0){
                                printf("iteración \t a \t b \t\t f(a) \t\t f(b) \t\t c \t\t f(c) \t\t error \n");
```

```
24
25                              while (i<=veces){
26                                      fa= x0+x1*a+x2*pow(a,2)+x3*pow(a,3);
27                                      fb= x0+x1*b+x2*pow(b,2)+x3*pow(b,3);
28                                      c = a-(fa*(b-a)/(fb-fa));
29                                      fc= x0+x1*c+x2*pow(c,2)+x3*pow(c,3);
30                                      error_calculado= fabs(fa-fc);
31                                      printf("%d \t %lf \t %lf \t %lf \t %lf \t %lf \t %lf \t %lf\n",i,a,b,fa,fb,c,fc,
                                                error_calculado);
32
33                                      raiz_calc= c;
34                                      if (fc*fa<0)
35                                              b= c;
36                                      else
37                                              a= c;
38                                      i++;
39                              }
40                              printf("la raíz cuadrada es %lf con un error calculado de %lf y error máximo de %lf",raiz_calc,
                                        error_calculado,error);
41                      }
42                      else
43                              printf("f(a) y f(b) no tienen signos diferentes");
44              }
45              else
46                      printf("El valor del error no es válido");
47      }
48      else
49              printf("Los valores del intervalo no inválidos");
50      return 0;
51  }
```

Debe usar estructuras algorítmicas iterativas con entrada controlada por contador y estructuras selectivas simples, dobles o anidadas.

No debe usar programación modular.