

Anthony Fabian Ramirez Orellana

Carne: 9490-22-958

Sección: "A"

Catedrático: Jorge Perez



Tarea 05 método secante.

16/03/2024

TAREA No 5

- 1) Utilice su implementación del método de La Secante (en C++) para encontrar una aproximación de la raíz de la siguiente función:

$$f(x) = \ln(x^2 + 1) - e^{0.4x} \cos(\pi x)$$

Con las siguientes condiciones:

- a) Calcular, con una tolerancia de 10^{-10} y 100 iteraciones máximo, la primera raíz negativa.
- b) Calcular, con una tolerancia de 10^{-10} y 100 iteraciones máximo, las primeras 4 raíces positivas.
- c) Calcular, con una tolerancia de 10^{-10} y 100 iteraciones máximo, la décima raíz positiva de la función.

Sugerencia: encuentre primero la gráfica de la función.

La salida de su programa debe considerar las siguientes columnas:

- Número de iteración
 - El valor de p_0 en cada iteración
 - El valor de p_1 en cada iteración
 - El valor de q_0 en cada iteración
 - El valor de q_1 en cada iteración
 - El valor de p calculado en cada iteración
 - El valor de $f(p)$ en cada iteración
 - El error absoluto en cada iteración: $(|(p-p_1)/p|)$
- 2) Comparar los resultados de los métodos Newton-Raphson y de La Secante (C++) utilizando las siguientes condiciones:

$$\text{Sea } f(x) = 2^x - 6\cos(x) = 0;$$

- a) Newton-Raphson: $p_0 = -6.5$; $TOL = 10^{-15}$; $IT = 100$;
- b) Secante: $p_0 = -6.5$; $p_1 = -5$; $TOL = 10^{-15}$; $IT = 100$;

Al final de su archivo debe incluir el código de su programa en C++.

No olvidar que el único formato aceptado para las tareas es PDF, cualquier otro formato no será calificado.

Código:

```

#include <iostream>//biblioteca estandar para operaciones de entrada/salida
(input/output stream)
#include <math.h>//archivo de cabecera que contiene funciones matematicas
basicas, es una libreria de terminos para uso de operaciones matematicas
#include <iomanip>//biblioteca se usa para ajustar decimales, esta libreria
la utilice unicamente para mostrar los 15 decimales de respuesta.

using namespace std;//importa todo el espacio de nombres std al codigo
actual, se utiliza mayormente para utilizar el prefijo std::

void menu(){//creo una funcion vacia o proceso, vacia ya que no quiero que
devuelva nada solo se ejecute, el proceso imprimira el menu para que se vea
bonito.
    cout<< "" << endl;//imprimo en la consola todo el menu, con saltos de
linea para que se vea bonito.
    cout<<"Ingrese la opcion que desea ejecutar:" << endl;//endl genera al
final de lo impreso anterior un salto de linea como un enter.

    cout<< "" << endl;//imprimo en la consola todo el menu, con saltos de
linea para que se vea bonito.

    cout<<"1)  $f(x) = \ln[(x^2) + 1] - [e^{(0.4x)}]\cos(\pi \cdot x) = 0$ " <<
endl;

    cout<< "" << endl;//imprimo en la consola todo el menu, con saltos de
linea para que se vea bonito.

    cout<<"2)  $f(x) = (2^x) - 6\cos(x) = 0$ " << endl;

    cout<< "" << endl;//imprimo en la consola todo el menu, con saltos de
linea para que se vea bonito.

    cout<<"3) Salir.";
}

void subMenu(){
    system("cls");//limpio la consola
    cout<<"Elija el valor con el que desea aproximar una solucion."<<endl;
    cout<< "" << endl;//imprimo en la consola todo el menu, con saltos de
linea para que se vea bonito.
    cout<<"1. TOL =  $10^{-10}$ , IT = 100, Primera raiz negativa."<<endl;
    cout<< "" << endl;//imprimo en la consola todo el menu, con saltos de
linea para que se vea bonito.
    cout<<"2. TOL =  $10^{-10}$ , IT = 100, Primera raiz positiva."<<endl;

```

```

        cout<< "" << endl; //imprimo en la consola todo el menu, con saltos de
        linea para que se vea bonito.
        cout<< "3. TOL = 10^-10, IT = 100, Segunda raiz positiva."<<endl;
        cout<< "" << endl; //imprimo en la consola todo el menu, con saltos de
        linea para que se vea bonito.
        cout<< "4. TOL = 10^-10, IT = 100, Tercera raiz positiva."<<endl;
        cout<< "" << endl; //imprimo en la consola todo el menu, con saltos de
        linea para que se vea bonito.
        cout<< "5. TOL = 10^-10, IT = 100, Cuarta raiz positiva."<<endl;
        cout<< "" << endl; //imprimo en la consola todo el menu, con saltos de
        linea para que se vea bonito.
        cout<< "6. TOL = 10^-10, IT = 100, Decima raiz positiva."<<endl;
        cout<< "" << endl; //imprimo en la consola todo el menu, con saltos de
        linea para que se vea bonito.
        cout<< "7. Regresar."<<endl;
    }

void subMenu2(){
    system("cls"); //limpio la consola
    cout<< "Elija el valor con el que desea aproximar una solucion."<<endl;
    cout<< "" << endl; //imprimo en la consola todo el menu, con saltos de
    linea para que se vea bonito.
    cout<< "1. Metodo Newton-Raphson TOL = 10^-15, IT = 100, p0 = -
6.5."<<endl;
    cout<< "" << endl; //imprimo en la consola todo el menu, con saltos de
    linea para que se vea bonito.
    cout<< "2. Metodo Secante TOL = 10^-15, IT = 100, p0 = -6.5, p1 = -
5."<<endl;
    cout<< "" << endl; //imprimo en la consola todo el menu, con saltos de
    linea para que se vea bonito.
    cout<< "3. Regresar."<<endl;
}

void cabezaTabla(){ //creo un proceso de tipo vacio, ya que no necesito que
me de una respuesta como una funcion, sino que realice una porcion de
codigo, enviandole a la propia funcion ningun parametro, ya que solo pintara
la consola.
    system("cls"); //codigo del cmd para limpiar la consola
    cout<< "-----
-----
-----" << endl; //cout se utiliza para imprimir en
consola el contenido dentro de los parentesis.
    cout<< "Iteracion          Numero p0          Numero
p1          Numero q0          Numero q1          Numero
P          f(p)          Error Absoluto          " << endl; //<< se puede

```

usar para concatenar texto a desplegar, endl es una intruccion de salto de linea.

```
        cout<<"-----  
-----  
-----"<< endl;  
}  
  
void contTabla(int ite, float nP0, float nP1, float nQ0, float nQ1, float  
nP, float nFp, float errAbs){//en este proceso si envio parametros para la  
correcta colocacion de los datos, pero igualmente es vacio para que se  
ejecute nada mas.  
    string espacio = "    ";  
    if(ite<=9){//un condicional if, que indica que si se cumple ite menor o  
igual a 9 realizara el contenido dentro del mismo, solo corre un espacio el  
codigo dependiendo si las iteraciones son de 1 o 2 digitos, ya que si no se  
veran desfazadas.  
        cout<<"-----  
-----  
-----"<< endl;  
        cout<<"  
"<<ite<<"          "<<fixed<<setprecision(15)<<nP0<<espacio<<fixed<<setprec  
ision(15)<<nP1<<espacio<<fixed<<setprecision(15)<<nQ0<<espacio<<fixed<<setpr  
ecision(15)<<nQ1<<espacio<<fixed<<setprecision(15)<<nP<<espacio<<fixed<<setp  
recision(15)<<nFp<<espacio<<fixed<<setprecision(15)<<errAbs<<endl;//fixed<<s  
etprecision(11) lo utilizo para dejar los parametros con 11 decimales  
    }  
    if(ite>9){//Este condicional es para todos los datos despues de la  
iteracion 9 ya que todo el contenido de la fila despues de la iteracion se  
debe correr un espacio a la izquierda para que no se vea desfazado.  
        cout<<"-----  
-----  
-----"<< endl;  
        cout<<" "<<ite<<"          "<<fixed<<setprecision(15)<<nP0<<espacio<  
<fixed<<setprecision(15)<<nP1<<espacio<<fixed<<setprecision(15)<<nQ0<<espaci  
o<<fixed<<setprecision(15)<<nQ1<<espacio<<fixed<<setprecision(15)<<nP<<espac  
io<<fixed<<setprecision(15)<<nFp<<espacio<<fixed<<setprecision(15)<<errAbs<<  
endl;  
    }  
}  
  
void cabezaTablaRapson(){//creo un proceso de tipo vacio, ya que no necesito  
que me de una respuesta como una funcion, sino que realice una porcion de  
codigo, enviandole a la propia funcion ningun parametro, ya que solo pintara  
la consola.  
    system("cls");//codigo del cmd para limpiar la consola
```

```

        cout<<"-----"
        -----"
endl; //cout se utiliza para imprimir en consola el contenido dentro de los
parentesis.
        cout<<"Iteracion          Numero p0          Numero fp0          Numero
p          f(p)          Error absoluto          "<< endl; //<< se
puede usar para concatenar texto a desplegar, endl es una instruccion de
salto de linea.
        cout<<"-----"
        -----"
    }

void contTablaRapson(int ite, float nP0, float nfP0, float np, float nfp,
float erAbs){
    if(ite<=9){ //un condicional if, que indica que si se cumple ite menor o
igual a 9 realizara el contenido dentro del mismo, solo corre un espacio el
codigo dependiendo si las iteraciones son de 1 o 2 digitos, ya que si no se
verandesfazadas.
        cout<<"-----"
        -----"
        cout<<"
"<<ite<<"          "<<fixed<<setprecision(15)<<nP0<<"          "<<fixed<<setprecis
ion(15)<<nfP0<<"          "<<fixed<<setprecision(15)<<np<<"          "<<fixed<<setprecisio
n(15)<<nfp<<"          "<<fixed<<setprecision(15)<<erAbs<<endl; //fixed<<setprecisio
n(11) lo utilizo para dejar los parametros con 11 decimales
    }
    if(ite>9){ //Este condicional es para todos los datos despues de la
iteracion 9 ya que todo el contenido de la fila despues de la iteracion se
debe correr un espacio a la izquierda para que no se vea desfazado.
        cout<<"-----"
        -----"
        cout<<" "<<ite<<"          "<<fixed<<setprecision(15)<<nP0<<"          "<<fi
xed<<setprecision(15)<<nfP0<<"          "<<fixed<<setprecision(15)<<np<<"          "<<fixe
d<<setprecision(15)<<nfp<<"          "<<fixed<<setprecision(15)<<erAbs<<endl;
    }
}

void mensajeExito(int i, float p, float fp){ // creo un proceso vacio solo
para mostrar el resultado correcto del metodo
    cout<<"Proceso finalizado exitosamente en la iteracion: "<< i << endl;
    cout<<"La solucion aproximada es p: "<<fixed<<setprecision(15)<< p <<
endl; //muestro el resultado usando fixed<<setprecision(15) para que el
resultado me lo muestre con 15 decimales de presicion.
    cout<<"Con f(p): "<<fixed<<setprecision(15)<< fp << endl;
    system("pause");
}

```

```

}

void mensajeFracaso(int i, float p, float fp){// creo un proceso vacio solo
para mostrar el resultado fallido del metodo
    cout<<"El metodo fracaso o procedimiento terminado sin exito en la
iteracion: "<< i << endl;
    cout<<"La solucion aproximada es p: "<<fixed<<setprecision(15)<< p <<
endl;//muestro el resultado usando fixed<<setprecision(15) para que el
resultado me lo muestre con 15 decimales de precision.
    cout<<"Con f(p): "<<fixed<<setprecision(15)<< fp <<
endl;
    system("pause");
}

int main(){//creo la funcion principal como int para que al final retorne 0 y
no tenga que declarar mas procesos para ejecutarlo
    int IT = 100, i = 1, opcion, secOpcion;//creo mis variable de tipo
entero
    float p0, fp0, p1, q0, q1, p, fp, dfp, errAb, TOL = pow(10, -10);//creo
mis variables de tipo float, que pueden contener muchos decimales
    float ayuda, ayuda2, ayuda3, ayuda4;//estas variables me ayudaran para
realizar los calculos, ya que si lo coloco de una en una linea me da un
resultado distinto

    while (opcion != 3)//creo un bucle while que se puede leer mientras que
opcion sea distinto de 3, realizara lo siguiente, este bucle me permite
mantenerme en el primer menu del programa
    {
        i = 2;//igualo nuevamente mi variable i a 1 para que en futuras
ejecuciones consecutivas siempre sea 1 al comienzo de cada bucle repetitivo
        menu();//llamo al proceso menu, que solo me imprime todas las
opciones del menu en consola

        if( (cin>>opcion).fail() ){//comprobo si la entrada de datos falla,
ya que el usuario puede ingresar un dato no valido, como una letra.
            system("cls");//de ser asi, limpio la consola
            cin.clear();//reseteo los flags(unos o mas bits que almacenan
valor binario o codigo)
            fflush(stdin);//limpio el buffer(espacio de memoria para
almacenar datos antes de procesarlos) de entrada
            opcion = 99;//igual la variable opcion a 99 para que se vaya al
caso default(por defecto) de mi condicional switch
        }
    }

```

```

        switch (opcion)//comienzo a validar la opcion que el usuario eligio
con un switch y la variable opcion
        {
            case 1://en el caso de devolver el numero 1 ejecuta lo siguiente
                TOL = pow(10, -10);//vuelvo a igualar la tolerancia ya que me
mantendré en este sub menu, porque para el otro necesito una tolerancia
distinta
                while (secOpcion != 7)//creo otro while, este se ejecutara
mientras la variable secOpcion sea distinta de 7
                {
                    subMenu();//Mando a llamar al proceso subMenu para que me
imprima en consola las opciones con las que cuento en este sub menu

                    if( (cin>>secOpcion).fail() ){//compruebo si la entrada de
datos falla, ya que el usuario puede ingresar un dato no valido, como una
letra.

                        system("cls");//de ser así, limpio la consola
                        cin.clear();//reseteo los flags(uno o mas bits que
almacenan valor binario o código)
                        fflush(stdin);//limpio el buffer(espacio de memoria para
almacenar datos antes de procesarlos) de entrada
                        secOpcion = 99;//igual la variable opcion a 99 para que
se vaya al caso default(por defecto) de mi condicional switch
                    }
                    switch (secOpcion)//comienzo a validar que numero digito
para la opcion
                    {
                        case 1://en caso de que haya sido el numero 1
                            system("cls");//utilizo esta linea para limpiar lo que
se mostro anteriormente en consola

                            cabezaTabla();//mando a llamar al proceso para que me
imprima la cabeza de la tabla, esta solo se debe de imprimir una vez, por
eso esta fuera de bucles de calculo

                            p0 = -0.5;//igualo mi primer valor al valor solicitado
                            p1 = 0;//igualo mi segundo valor al valor solicitado
                            q0 = ( log( (pow(p0, 2) + 1) ) - ( exp(0.4*p0) *
cos(M_PI*p0) ) );//realizo el calculo de mi funcion metiendo el valor de
p0, aqui si me permite poner todo en una linea
                            q1 = ( log( (pow(p1, 2) + 1) ) - ( exp(0.4*p1) *
cos(M_PI*p1) ) );//realizo el calculo de mi funcion metiendo el valor de
p1

                            while (i <= IT)//comienzo mi bucle de calculos, mientras
i sea menor a IT osea en menos de 100 repeticiones

```



```

        {
            ayuda = (q1 * ( p1-p0 ) );//Utilizo mi primer
            variable de ayuda, esta me permite tener un valor mas exacto a la hora de
            realizar el calculo

            ayuda2 = q1-q0; //utilizo mi segunda variable de
            ayuda, esta me permite tener el valor correcto a la hora de que lo calcula
            mi PC

            p = ( p1 - ( ayuda / ayuda2 ) );//unifico mis
            variables para que se operen y me entreguen el valor "p"

            fp = ( log( (pow(p, 2) + 1) ) - ( exp(0.4*p) *
            cos(M_PI*p) ) );//valido mi valor p en la funcion para corroborar que sea
            cercano a cero

            errAb = abs( (p-p1)/p );//uso mi variable errAb para
            igualarla al calculo de mi error

            if (errAb < TOL)//Utilizo un condicional que me
            valida si el error es menor a la tolerancia para acabar el bucle y dar una
            respuesta exitosa mas rapida
            {
                contTabla(i, p0, p1, q0, q1, p, fp,
            errAb);//llamo al proceso enviandole todos los parametros que quiero en la
            tabla para que los imprima en consola de la forma mas ordenada ya
            anteriormente configurada

                mensajeExito(i, p, fp);//llamo mi proceso para
            que me imprima el mensaje de exito con los valores que le envio

                break;//uso el break apra salir del bucle y
            poder proseguir con el programa y poder seguir utilizandolo
            }

            contTabla(i, p0, p1, q0, q1, p, fp, errAb);//si se
            salta el if entonces se imprime la tabla con esta llamada del proceso
            i += 1;//sumo 1 a mi variable i para que el bucle no
            sea infinito

            p0 = p1;//igualo p0 a p1 para realizar el siguiente
            calculo de bucle

            q0 = q1;//igualo q0 a q1 para realizar el siguiente
            calculo de bucle

            p1 = p;//igualo p1 a p para realizar el siguiente
            calculo de bucle

            q1 = fp;//igualo q1 a fp para realizar el siguiente
            calculo de bucle
        }

        if(i >= 99){//valido que mi bucle haya salido sin una
            respuesta previa, ya que si lo dejo sin if se darian 2 respuestas

            mensajeFracaso(i, p, fp);//llamo a mi proceso de
            mensaje de fracaso mandando los valores que deseo que me muestre

```

```

    }
    i = 2; //igualo la variable i a 1 para seguir utilizando
el programa de forma segura
    break; //acaba el caso 1, a partir de aqui hasta el siguiente
menu principal es lo mismo solo cambiando los valores
    case 2:
        system("cls");

        cabezaTabla();

        p0 = 0;
        p1 = 0.5;
        q0 = ( log( (pow(p0, 2) + 1) ) - ( exp(0.4*p0) *
cos(M_PI*p0) ) );
        q1 = ( log( (pow(p1, 2) + 1) ) - ( exp(0.4*p1) *
cos(M_PI*p1) ) );
        while (i <= IT)
        {
            ayuda = (q1 * ( p1-p0 ) );
            ayuda2 = q1-q0;
            p = ( p1 - ( ayuda / ayuda2 ) );
            fp = ( log( (pow(p, 2) + 1) ) - ( exp(0.4*p) *
cos(M_PI*p) ) );
            errAb = abs( (p-p1)/p );

            if (errAb < TOL)
            {
                contTabla(i, p0, p1, q0, q1, p, fp, errAb);
                mensajeExito(i, p, fp);
                break;
            }
            contTabla(i, p0, p1, q0, q1, p, fp, errAb);
            i += 1;
            p0 = p1;
            q0 = q1;
            p1 = p;
            q1 = fp;
        }
        if(i >= 99){
            mensajeFracaso(i, p, fp);
        }
        i = 2;
        break;
    case 3:
        system("cls");

```

```

        cabezaTabla();

        p0 = 1.5;
        p1 = 2;
        q0 = ( log( (pow(p0, 2) + 1) ) - ( exp(0.4*p0) *
cos(M_PI*p0) ) );
        q1 = ( log( (pow(p1, 2) + 1) ) - ( exp(0.4*p1) *
cos(M_PI*p1) ) );
        while (i <= IT)
        {
            ayuda = (q1 * ( p1-p0 ) );
            ayuda2 = q1-q0;
            p = ( p1 - ( ayuda / ayuda2 ) );
            fp = ( log( (pow(p, 2) + 1) ) - ( exp(0.4*p) *
cos(M_PI*p) ) );

            errAb = abs( (p-p1)/p );

            if (errAb < TOL)
            {
                contTabla(i, p0, p1, q0, q1, p, fp, errAb);
                mensajeExito(i, p, fp);
                break;
            }
            contTabla(i, p0, p1, q0, q1, p, fp, errAb);
            i += 1;
            p0 = p1;
            q0 = q1;
            p1 = p;
            q1 = fp;
        }
        if(i >= 99){
            mensajeFracaso(i, p, fp);
        }
        i = 2;
    break;
    case 4:
        system("cls");

        cabezaTabla();

        p0 = 2;
        p1 = 2.5;
        q0 = ( log( (pow(p0, 2) + 1) ) - ( exp(0.4*p0) *
cos(M_PI*p0) ) );

```

```

        q1 = ( log( (pow(p1, 2) + 1) ) - ( exp(0.4*p1) *
cos(M_PI*p1) ) );
        while (i <= IT)
        {
            ayuda = (q1 * ( p1-p0 ) );
            ayuda2 = q1-q0;
            p = ( p1 - ( ayuda / ayuda2 ) );
            fp = ( log( (pow(p, 2) + 1) ) - ( exp(0.4*p) *
cos(M_PI*p) ) );

            errAb = abs( (p-p1)/p );

            if (errAb < TOL)
            {
                contTabla(i, p0, p1, q0, q1, p, fp, errAb);
                mensajeExito(i, p, fp);
                break;
            }
            contTabla(i, p0, p1, q0, q1, p, fp, errAb);
            i += 1;
            p0 = p1;
            q0 = q1;
            p1 = p;
            q1 = fp;
        }
        if(i >= 99){
            mensajeFracaso(i, p, fp);
        }
        i = 2;
    break;
    case 5:
        system("cls");

        cabezaTabla();

        p0 = 3.5;
        p1 = 4;
        q0 = ( log( (pow(p0, 2) + 1) ) - ( exp(0.4*p0) *
cos(M_PI*p0) ) );
        q1 = ( log( (pow(p1, 2) + 1) ) - ( exp(0.4*p1) *
cos(M_PI*p1) ) );
        while (i <= IT)
        {
            ayuda = (q1 * ( p1-p0 ) );
            ayuda2 = q1-q0;
            p = ( p1 - ( ayuda / ayuda2 ) );

```

```

cos(M_PI*p) ) );
fp = ( log( (pow(p, 2) + 1) ) - ( exp(0.4*p) *
errAb = abs( (p-p1)/p );

if (errAb < TOL)
{
    contTabla(i, p0, p1, q0, q1, p, fp, errAb);
    mensajeExito(i, p, fp);
    break;
}
contTabla(i, p0, p1, q0, q1, p, fp, errAb);
i += 1;
p0 = p1;
q0 = q1;
p1 = p;
q1 = fp;
}
if(i >= 99){
    mensajeFracaso(i, p, fp);
}
i = 2;
break;
case 6:
    system("cls");

    cabezaTabla();

    p0 = 9.5;
    p1 = 10;
q0 = ( log( (pow(p0, 2) + 1) ) - ( exp(0.4*p0) *
cos(M_PI*p0) ) );
q1 = ( log( (pow(p1, 2) + 1) ) - ( exp(0.4*p1) *
cos(M_PI*p1) ) );
while (i <= IT)
{
    ayuda = (q1 * ( p1-p0 ) );
    ayuda2 = q1-q0;
    p = ( p1 - ( ayuda / ayuda2 ) );
    fp = ( log( (pow(p, 2) + 1) ) - ( exp(0.4*p) *
cos(M_PI*p) ) );
    errAb = abs( (p-p1)/p );

    if (errAb < TOL)
    {
        contTabla(i, p0, p1, q0, q1, p, fp, errAb);

```

```

        mensajeExito(i, p, fp);
        break;
    }
    contTabla(i, p0, p1, q0, q1, p, fp, errAb);
    i += 1;
    p0 = p1;
    q0 = q1;
    p1 = p;
    q1 = fp;
}
if(i >= 99){
    mensajeFracaso(i, p, fp);
}
i = 2;
break;
default://utilizo el caso default para cualquier otro numero
que no este declarado ni mostrado en el menu
    system("cls");//limpio la pantalla
    cout<< "Porfavor ingrese una opcion valida del menu."<<
endl;//muestra el mensaje de que debe ingresar una opcion valida
    break;
}
}
break;
case 2://inicio del menu en caso 2
    i = 1;
    TOL = pow(10, -15);//cambio mi tolerancia a 10*-15 para tener
una tolerancia distinta segun el insiso de la tarea
    while (secOpcion != 3)//comienzo mi while para mantenerme dentro
del sub meni hasta que el valor se secOpcion sea 3
    {
        subMenu2();//llamo a submenu2 para que me imprima el menu en
consola

        if( (cin>>secOpcion).fail() ){//compurebo si la entrada de
datos falla, ya que el usuario puede ingresar un dato no valido, como una
letra.

            system("cls");//de ser asi, limpio la consola
            cin.clear();//reseteo los flags(uno o mas bits que
almacenan valor binario o codigo)
            fflush(stdin);//limpio el buffer(espacio de memoria para
almacenar datos antes de procesarlos) de entrada
            secOpcion = 99;//igual la variable opcion a 99 para que
se vaya al caso default(por defecto) de mi condicional switch
        }
    }

```

```

        switch (secOpcion)//comienzo a validar el numero que
introdujo el usuario
    {
        case 1://si introdujo el 1 se ejecuta el siguiente caso que
es el Newton-Raphson
            i = 1;
            system("cls");//limpio la consola

            cabezaTablaRapson();//llamo a mi subprocesso aislado para
este inciso

            p0 = -6.5;//utilizo solo la variable de p0 para poner el
valor dado del inciso

            while (i <= IT)//mi bucle de calculos que esta mientras
i sea menor o igual a IT para solo dar 100 iteraciones
            {
                fp0 = ( pow(2, p0) - 6 * cos(p0) );//introduzco
el valor de p0 en mi funcion para comenzar con los calculos
                ayuda = log(2);//utilizo una variable de ayuda por
si el valor de log(2) me genera incompatibilidad
                ayuda2 = ( ayuda * (pow(2, p0)) );//utilizo la
variable ayuda2 para multiplicar el logaritmo con 2 a la potencia de p0 ya
que puede generarme un valor erroneo
                dfp = ( ayuda2 + (6 * sin(p0)) );//introduzco el
valor de p0 en la derivada de mi funcion
                p = ( p0 - (fp0/dfp) );//realizo el calculo de p
                fp = ( pow(2, p) - 6 * cos(p) );//introduzco el
valor de p en mi funcion para desplegarlo y ver que tan cercano a cero se
esta volviendo

                errAb = abs( (p-p0) / p );//igualo mi variable
errAb al calculo de mi error

                if(errAb < TOL){//valido si mi error es menor a la
tolerancia para dar una respuesta mas rapida
                    contTablaRapson(i, p0, fp0, p, fp,
errAb);//llamo a mi proceso para desplegar la tabla con los valores mandados

                    mensajeExito(i, p, fp0);//llamo mi proceso para
desplegar el mensaje de exito con mis valores calculados
                    break;//uso el bracke para no seguir iterando
                }
                contTablaRapson(i, p0, fp0, p, fp, errAb);//si se
ignora el if anterior entonces despliego la informacion con esta llamada de
proceso

```

```

        i += 1; //sumo uno a mi variable i para que haya una
salida del bucle
        p0 = p; //igualo p0 a p para seguir operando
    }
    if(i >= 99){ //valido si mi bucle anterior salio por el
limite de iteraciones sea cumplido y despliegue mi mensaje de erro
        mensajeFracaso(i, p, fp); //el mensaje de error se
despliega con los datos que mando a este proceso
    }
    i = 1; //igualo la variable a 1 para poder seguir usando
el programa desde este sub menu
    break;
    case 2: //inicio el caso 2 de la Secante, lo unico que
cambia aqui son los valores y el calculo de las funciones ya que la
funciones la misma que el sub menu anterior para poder igualarlos
        i = 2;
        system("cls");

        cabezaTabla();

        p0 = -6.5;
        p1 = -5;
        q0 = ( pow(3, p0)) - (6*cos(p0)) );
        q1 = ( pow(3, p1)) - (6*cos(p1)) );
        while (i <= IT)
        {
            ayuda = (q1 * ( p1-p0 ) );
            ayuda2 = q1-q0;
            p = ( p1 - ( ayuda / ayuda2 ) );
            fp = ( pow(3, p)) - (6*cos(p)) );
            errAb = abs( (p-p1)/p );

            if (errAb < TOL)
            {
                contTabla(i, p0, p1, q0, q1, p, fp, errAb);
                mensajeExito(i, p, fp);
                break;
            }
            contTabla(i, p0, p1, q0, q1, p, fp, errAb);
            i += 1;
            p0 = p1;
            q0 = q1;
            p1 = p;
            q1 = fp;
        }

```



```

        if(i >= 99){
            mensajeFracaso(i, p, fp);
        }
        i = 2;
        break;
        default://utilizo el caso deafult para validar si el valor
ingresado no esta en el submenu
            system("cls");
            cout<< "Porfavor ingrese una opcion valida del menu."<<
endl;//muestra el mensaje de que debe ingresar una opcion valida
            break;
        }
    }
    break;
    default://utilizo el caso deafult para validar si el valor ingresado
no esta en el submenu
        system("cls");
        cout<< "Porfavor ingrese una opcion valida del menu."<<
endl;//muestra el mensaje de que debe ingresar una opcion valida
        break;
    }
    secOpcion = 0;//igualo la variable secOpcion a cero para poderseguir
utilizando el programa desde este submenu
}

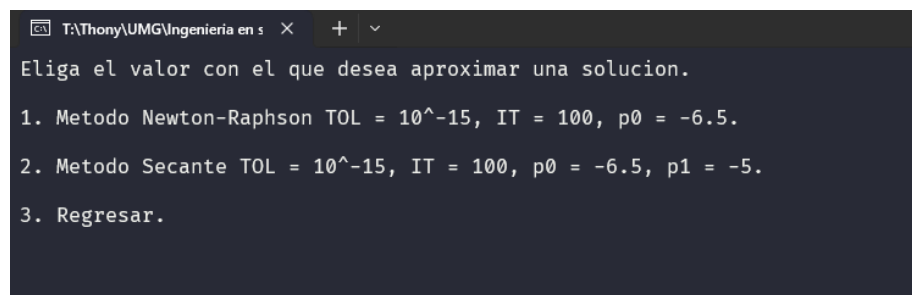
return 0;//regreso cero para cerrar de una vez el programa
}

```

Comparación me

todo Newton-Raphons vs Secante:

Comenzando en el submenú para comparar el resultado de los métodos.



```

T:\Thony\UMG\Ingeniería en s  ×  +  ▾
Eliga el valor con el que desea aproximar una solucion.
1. Metodo Newton-Raphson TOL = 10^-15, IT = 100, p0 = -6.5.
2. Metodo Secante TOL = 10^-15, IT = 100, p0 = -6.5, p1 = -5.
3. Regresar.

```

Utilizamos primero la opción del método Newton-Raphson, encontrando que le tomo únicamente 4 iteraciones para calcular la raíz, y encontró la raíz que se encuentra por -10 .

Iteracion	Numero p0	Numero fp0	Numero p	f(p)	Error absoluto
1	-6.500000000000000	-5.848477363586426	-11.058219909667969	-0.375158965587616	0.412201970815659
2	-11.058219909667969	-0.375158965587616	-10.995573997497559	0.000491521961521	0.005697375163436
3	-10.995573997497559	0.000491521961521	-10.995656013488770	-0.000000601846978	0.000007458944765
4	-10.995656013488770	-0.000000601846978	-10.995656013488770	-0.000000601846978	0.000000000000000
Proceso finalizado exitosamente en la iteracion: 4					
La solución aproximada es p: -10.995656013488770					
Con f(p): -0.000000601846978					
Presione una tecla para continuar . . .					

En cambio comparando con el metodo secante le toma 6 iteraciones encontrar una raiz y esta vez encuentra la raiz mas cercana al intervalo ingresado, se podria decir que el anterior es mas rapido pero no da el valor mas cercano, en cambio esta se toma mas proceso pero da la mas cercana.

Iteracion	Numero p0	Numero p1	Numero q0	Numero q1	Numero P	f(p)	Error Absoluto
2	-6.500000000000000	-5.000000000000000	-5.858733654022217	-1.697857975959778	-4.387920379638672	1.920892715454102	0.139491960406303
3	-5.000000000000000	-4.387920379638672	-1.697857975959778	1.920892715454102	-4.712822437286377	0.003040987998247	0.068940013647079
4	-4.387920379638672	-4.712822437286377	1.920892715454102	0.003040987998247	-4.713337421417236	-0.000052107116062	0.000109261032776
5	-4.712822437286377	-4.713337421417236	0.003040987998247	-0.000052107116062	-4.713328838348389	-0.00000555533973	0.000001821020533
6	-4.713337421417236	-4.713328838348389	-0.000052107116062	-0.00000555533973	-4.713328838348389	-0.00000555533973	0.000000000000000
Proceso finalizado exitosamente en la iteracion: 6							
La solución aproximada es p: -4.713328838348389							
Con f(p): -0.00000555533973							
Presione una tecla para continuar . . .							

Video del funcionamiento:

https://drive.google.com/file/d/1GjMaoMqtlpN1c4xZv7-uVMg8zFY7AEN5/view?usp=drive_link