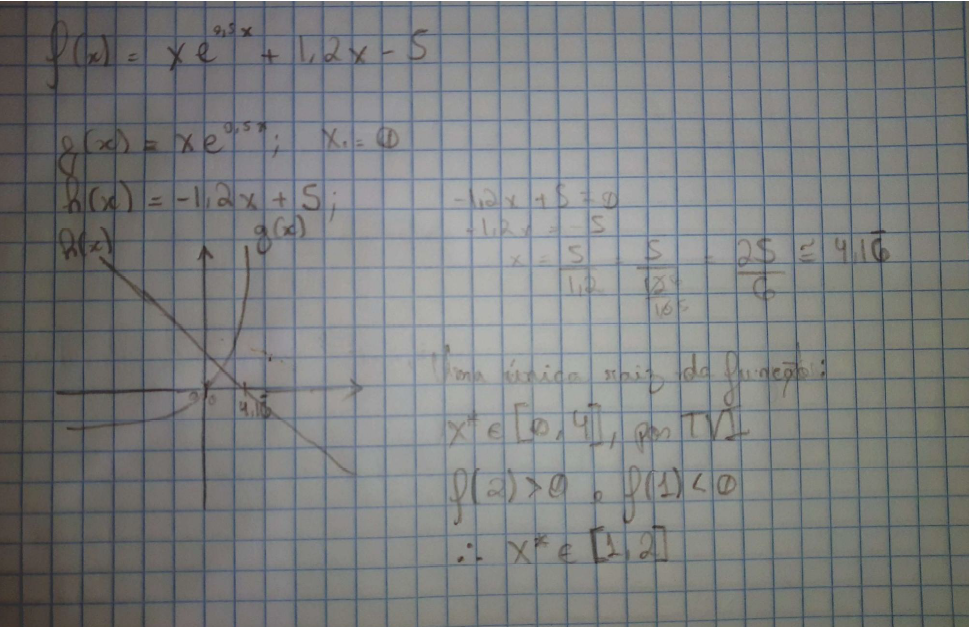
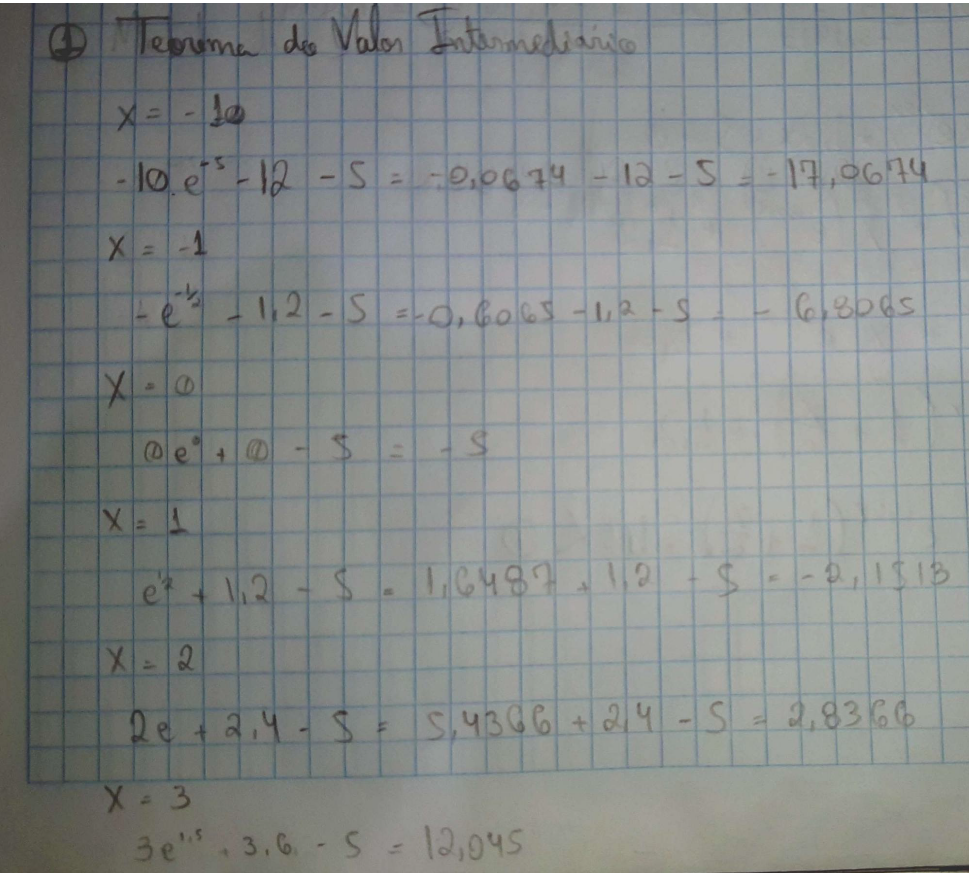
Trabalho de Cálculo Numérico

- Guilherme Martiniano de Oliveira - 11215765

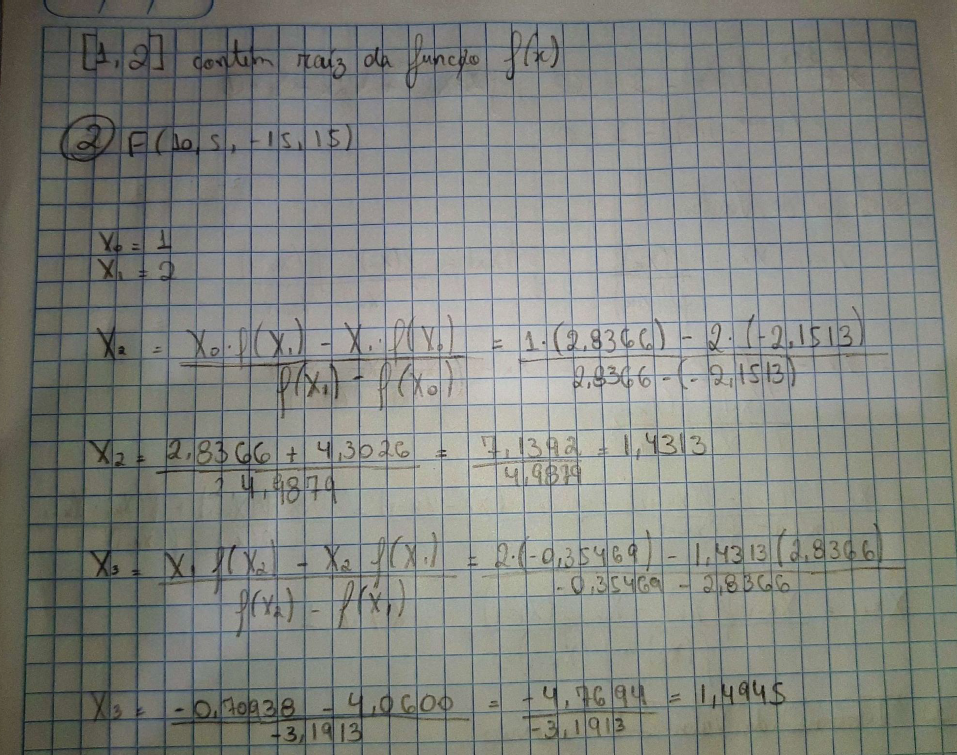
- Gustavo Fernandes Carneiro de Castro - 11369684

- Mateus Miquelino da Silva - 11208412

Item a)



Item b)



Item c)

// Guilherme Martiniano de Oliveira 11215765

// Gustavo Fernandes Carneiro de Castro 11369684

// Mateus Miquelino da Silva 11208412

#include <stdio.h>

#include <math.h>

void metodoSecanteFloat(float X0, float E); // Metodo da Secante utilizando X0 que o usuario escolhe, com isso o X1 necessario da formula sera X0 + 0.2

float funcaoFloat(float X); // Funcao f(x) que deseja ser estudada para encontrar a raiz

void metodoSecanteDouble(double X0, double E); // Metodo da Secante utilizando X0 que o usuario escolhe, com isso o X1 necessario da formula sera X0 + 0.2

double funcaoDouble(double X); // Funcao f(x) que deseja ser estudada para encontrar a raiz

float EULER = 2.718281828459045235360287; // Definicao do Numero de Euler pego de: https://pt.wikipedia.org/wiki/E\_(constante\_matem%C3%A1tica)

int main(){

float X0f, Ef;

double X0d, Ed;

printf("\nEntre com o X0 Float: ");

scanf("%f", &X0f);

printf("\nEntre com a precisao desejada: ");

scanf("%f", &Ef);

metodoSecanteFloat(X0f, Ef);

printf("\nEntre com o X0 Double: ");

scanf("%lf", &X0d);

printf("\nEntre com a precisao desejada: ");

scanf("%lf", &Ed);

metodoSecanteDouble(X0d, Ed);

}

void metodoSecanteFloat( float X0, float E ){

float X, Xk, Xk1, Xk2;

int i = 0;

Xk = X0;

Xk1 = X0 + 0.2;

while(!(fabs(Xk1 - Xk) < E) && i < 20){

Xk2 = ((Xk \* funcaoFloat(Xk1)) - (Xk1 \* funcaoFloat(Xk))) / (funcaoFloat(Xk1) - funcaoFloat(Xk));

Xk = Xk1;

Xk1 = Xk2;

i++;

}

if( i == 20 ){

printf("Seu Xo nao foi proximo o suficiente da raiz da funcao.\n");

}else{

X = Xk1;

printf("A aproximacao da raiz eh: %f\n", X);

}

}

void metodoSecanteDouble(double X0, double E)

{

double X, Xk, Xk1, Xk2;

int i = 0;

Xk1 = X0;

Xk1 = X0 + 0.2;

while(!(fabs(Xk1 - Xk) < E) && i < 20)

{

Xk2 = ((Xk \* funcaoDouble(Xk1)) - (Xk1 \* funcaoDouble(Xk))) / (funcaoDouble(Xk1) - funcaoDouble(Xk));

Xk = Xk1;

Xk1 = Xk2;

i++;

}

if( i == 20 ){

printf("Seu X0 nao foi proximo o suficiente da raiz da funcao.\n");

}else{

X = Xk1;

printf("A aproximacao da raiz e: %lf\n", X);

}

}

float funcaoFloat( float X ){

float Y;

Y = (X \* pow(EULER, 0.5 \* X)) + (1.2 \* X) - 5;

return (Y);

}

double funcaoDouble(double X)

{

double Y;

Y = (X \* pow(EULER, 0.5 \* X)) + (1.2 \* X) - 5;

return (Y);

}

Resultados:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Float | | | Double | | |
| X0 | Precisão | Raiz Aprox. | X0 | Precisão | Raiz Aprox. |
| 1 | 0.01 | 1.504925 | 1 | 0.01 | 1.504990 |
| 1 | 0.001 | 1.504988 | 1 | 0.001 | 1.504990 |
| 1 | 0.0001 | 1.504988 | 1 | 0.0001 | 1.504988 |
| 3 | 0.0001 | 1.504988 | 1 | 0.0001 | 1.504988 |
| 100 | 0.0001 | Não Existe (Passou de 20 Iterações) | 3 | 0.0001 | 0.000000  (Erro) |

Conclusão:

Após análises feitas, a equação linear possuía apenas uma raiz real. Depois, foi calculada essa raiz por intermédio do método da secante. No item b, o resultado estava se aproximando da raiz (1.504988), mesmo após apenas 2 iterações. No item c, através do uso do computador, aumentamos o número de iterações máximas para 20, e, ao selecionar uma precisão menor que 0.01, o resultado do método foi igual à raiz. Porém, nas análises com uma precisão maior, o resultado se afastou um pouco do esperado, principalmente no float, considerando sua limitação de memória, conforme visualizado na imagem abaixo. Além disso, Ao afastar muito o X0, o método ficou inutilizável.

