

I. PARCIAL NO.1

A. Juan José Loaiza Lujan.

Universidad del Valle sede Zarzal, Valle del Cauca, Colombia.

juan.loaiza.lujan@correounivalle.edu.co

Programa: Algoritmia y programación

Profesor: Paul Angarita Jimenez

```

1 #include <cstdlib>
2 #include <stdio.h>
3 using namespace std;
4 int main () {
5     int num1, num2;
6     num1 = 230;
7     num2 = 345;
8     int div1 = 0;
9     int div2 = 0;
10    for (int i = 1; i <= num1/2; i++) {
11        if (num1 % i == 0) div1 += i;
12    }
13    for (int i = 1; i <= num2/2; i++) {
14        if (num2 % i == 0) div2 += i;
15    }
16    if (num1 == div2 && num2 == div1) printf("%d y %d son amigos ", num1, num2);
17    else printf("%d y %d no son amigos ", num1, num2);
18 }

```

SALIDA

230 y 345 no son amigos

Fig. 1. Identificador de números amigos

```

1 #include <iostream> //diva
2 using namespace std;
3 int main(){
4     double a00,a01,a02,a10,a11,a12,a20,a21,a22,total;
5     cout<<"Ingrese los valores:"<<endl;
6     cout<<"a00 a01 a02" <<endl;
7     cout<<"a10 a11 a12" <<endl;
8     cout<<"a20 a21 a22" <<endl;
9     cout<<"a00: ";
10    cin>>a00;
11    cout<<"a01: ";
12    cin>>a01;
13    cout<<"a02: ";
14    cin>>a02;
15    cout<<"a10: ";
16    cin>>a10;
17    cout<<"a11: ";
18    cin>>a11;
19    cout<<"a12: ";
20    cin>>a12;
21    cout<<"a20: ";
22    cin>>a20;
23    cout<<"a21: ";
24    cin>>a21;
25    cout<<"a22: ";
26    cin>>a22;
27    /*
28     1 1a00 a10 a20 12a00 12a10
29     1a01 a11 a21 12a01 12a11
30     1a02 a12 a22 12a02 12a12
31     */
32    total=a00*a11*a22 + a10*a21*a02 +a20*a01*a12;
33    total=total-(a02*a11*a20)*-1 + (a12*a21*a00)*-1 + (a20
34    if(total!=0)
35    {
36        //...

```

Fig. 2. Primera parte de la Matris inversa en c++

```

1 #include <iostream> //diva
2 using namespace std;
3 int main(){
4     double a00,a01,a02,a10,a11,a12,a20,a21,a22,total;
5     cout<<"Ingrese los valores:"<<endl;
6     cout<<"a00 a01 a02" <<endl;
7     cout<<"a10 a11 a12" <<endl;
8     cout<<"a20 a21 a22" <<endl;
9     cout<<"a00: ";
10    cin>>a00;
11    cout<<"a01: ";
12    cin>>a01;
13    cout<<"a02: ";
14    cin>>a02;
15    cout<<"a10: ";
16    cin>>a10;
17    cout<<"a11: ";
18    cin>>a11;
19    cout<<"a12: ";
20    cin>>a12;
21    cout<<"a20: ";
22    cin>>a20;
23    cout<<"a21: ";
24    cin>>a21;
25    cout<<"a22: ";
26    cin>>a22;
27    /*
28     1 1a00 a10 a20 12a00 12a10
29     1a01 a11 a21 12a01 12a11
30     1a02 a12 a22 12a02 12a12
31     */
32    total=a00*a11*a22 + a10*a21*a02 +a20*a01*a12;
33    total=total-(a02*a11*a20)*-1 + (a12*a21*a00)*-1 + (a20
34    if(total!=0)
35    {
36        //...

```

Fig. 3. Segunda parte de la matris inversa en c++

```

Ingrese los valores:
a00 a01 a02
a10 a11 a12
a20 a21 a22
a00: 1
a01: 2
a02: 3
a10: 4
a11: 5
a12: 6
a20: 7
a21: 8
a22: 9
a00: 0.399999 0.799999 0.799999
a01: 0.399999 0.799999 0.799999
a02: 0.399999 0.799999 0.799999
a10: 0.399999 0.799999 0.799999
a11: 0.399999 0.799999 0.799999
a12: 0.399999 0.799999 0.799999
a20: 0.399999 0.799999 0.799999
a21: 0.399999 0.799999 0.799999
a22: 0.399999 0.799999 0.799999
Presione una tecla para continuar

```

Fig. 4. Prueba de la Matris

```

1 #include <iostream>
2 #include <math.h>
3 #include <conio.h>
4
5 int IntMaximosIntentos,intentos=0;
6 float f(floatRaizInicial,xn=0,fdeRn=0,DerivadaDeRn=0,error=0,tolerancia=0.001, valorvalt, v=0);
7
8
9 int main()
10 {
11     int a, b;
12     float RaizInicial;
13     cout<<"Resistencia\n";
14     scanf("%f", &fRaizInicial);
15     printf("Voltaje\n");
16     scanf("%f", &valorvalt);
17     printf("Ingrese el valor del numero maximo de intentos:\n");
18     scanf("%d", &IntMaximosIntentos);
19     RaizInicial=fRaizInicial;
20     valorvalt=valorvalt;
21     do{
22         //...

```

Fig. 5. Primera parte del metodo de newton

```

34 scanf("%d", &IntMaximosIntentos);
35
36 Xn=floatRaizInicial;
37 V=valorVall;
38
39 do{
40     //Aqui puede depositar su función --- f(x) --- si la va a cambiar.
41     fdeKn=(V-0.7)*(V-0.7)/Xn;
42
43     //Aqui puede depositar la derivada de su función --- f'(x) --- si la va a cambiar.
44     DerivadafdeKn=2*(V-0.7)/Xn;
45
46     //Este es el algoritmo principal
47     Xnpl=(V-0.7)-(fdeKn/DerivadafdeKn);
48     //Este es el algoritmo principal
49
50     error=(V-0.7)-Xnpl;
51
52     if(error<0)
53     error=Xnpl-(V-0.7);
54     intentos++;
55     if(intentos==IntMaximosIntentos)
56     error=tolerancia;
57     V=Xnpl;
58 }while(error>tolerancia);
59
60 printf("La raíz mas aproximada es %f, con %d intentos", Xnpl,intentos);
61 getch();
62 }
63
64
65
66

```

Fig. 6. Segunda parte del metodo de newton

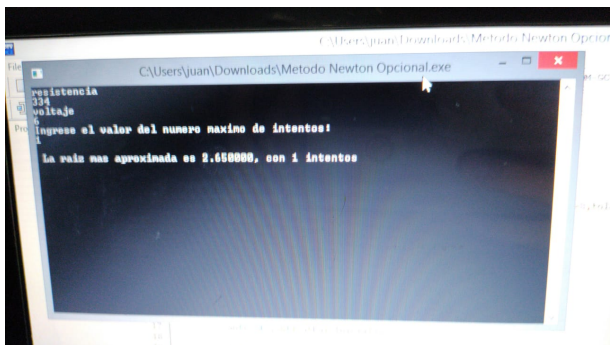


Fig. 7. Prueba de newton