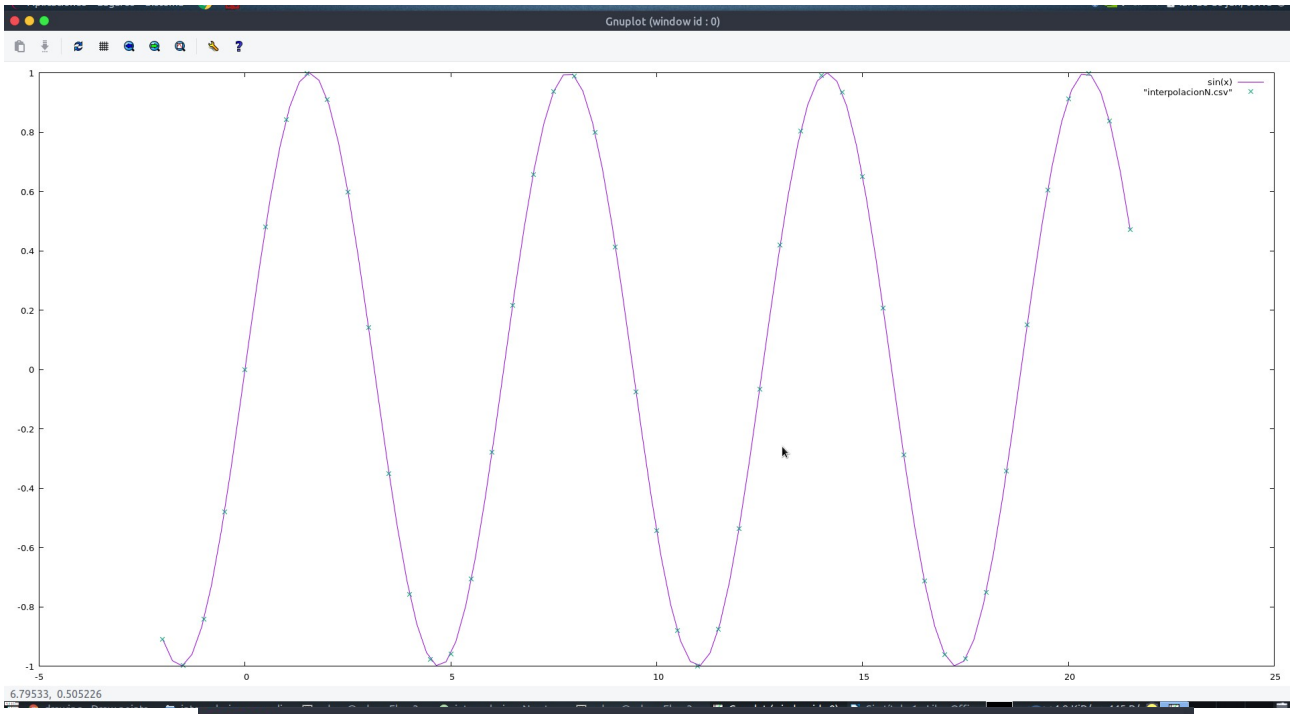
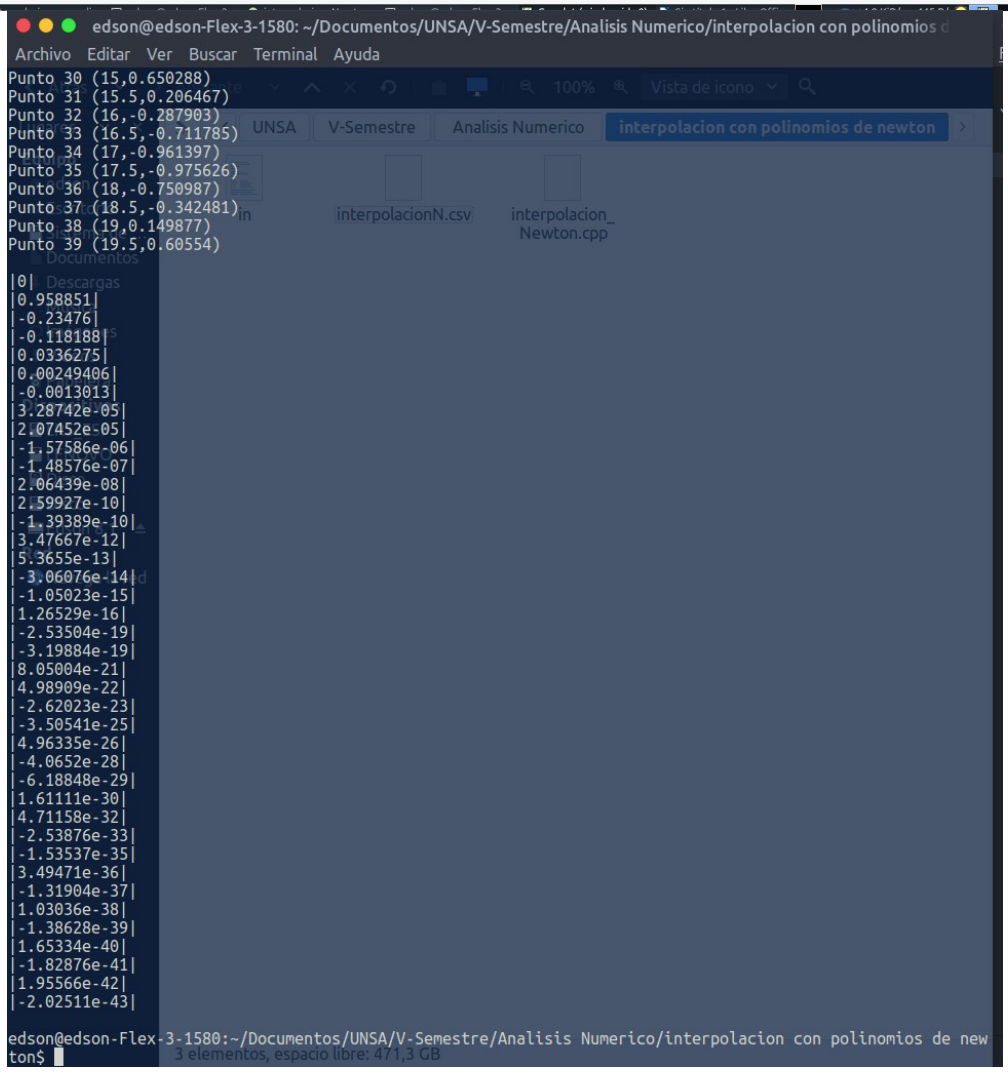
Nombre : Edson Victor Lipa Urbina

interpolacion con polinomios de newton con un ejemplo con la funcion seno “sin(x)” con 40 puntos desde 0 a 20 con intervalos de 0.5

y con evalucion con puntos de -2 a 22





Codigo:

```
#include <iostream>
#include <cmath>
#include <map>
#include <vector>
#include <fstream>
#include <stdlib.h>

using namespace std;

typedef long double number;
typedef vector<number> row;
typedef vector<row> matrix;
typedef vector<map<char,number>> lista;
typedef pair<vector<number>,lista> funcion;

void llenarcoordenadas(lista &coordenadas)
{
	std::map<char, number> temp;

	temp['x']=1;
	temp['y']=2*cos((3.14159/2)*1);
	coordenadas.push_back(temp);
	temp['x']=2;
	temp['y']=2*cos((3.14159/2)*2);
	coordenadas.push_back(temp);
	temp['x']=3;
	temp['y']=2*cos((3.14159/2)*3);
	coordenadas.push_back(temp);
	temp['x']=4;
	temp['y']=2*cos((3.14159/2)*4);
	coordenadas.push_back(temp);
	temp['x']=5;
	temp['y']=2*cos((3.14159/2)*5);
	coordenadas.push_back(temp);
	temp['x']=6;
	temp['y']=2*cos((3.14159/2)*6);
	coordenadas.push_back(temp);
	temp['x']=7;
	temp['y']=2*cos((3.14159/2)*7);
	coordenadas.push_back(temp);
	temp['x']=8;
	temp['y']=2*cos((3.14159/2)*8);
	coordenadas.push_back(temp);

	cout<<"\tPuntos iniciales dados"<<endl;
 for (int i = 0; i < coordenadas.size(); ++i)
 {

	cout<<"Punto "<<i+1<<" ("<<coordenadas[i]['x']<<","<<coordenadas[i]['y']<<")"<<endl;
 }
 cout<<endl;
}
void llenarcoordenadas(lista &coordenadas,float desde,float hasta,float inter)
{
	coordenadas[0]['x']=0;
	coordenadas[0]['y']=sin(0);
	int cont=0;
	for (float i = desde; i < hasta; i=i+inter) {

		coordenadas[cont]['x']=i;
```

```
		std::cout << "/* message */" <<i<< '\n';
		coordenadas[cont]['y']=sin(coordenadas[cont]['x']);
		cont++;
	}

	cout<<"\tPuntos iniciales dados"<<endl;
 for (int i = 0; i < coordenadas.size(); ++i)
 {

	cout<<"Punto "<<i<<" ("<<coordenadas[i]['x']<<","<<coordenadas[i]['y']<<")"<<endl;
 }
 cout<<endl;
}

void print(matrix m)
{

	for (unsigned i = 0; i < m.size(); ++i)
	{
		for (unsigned j = 0; j < m[0].size(); ++j)
		{
			if(j==0){cout<<"|";}
			if(j!=0){cout<<"\t";}
			cout<<m[i][j];
		}
		cout<<"|"<<endl;
	}
	cout<<endl;
}
void print(row &m)
{
	unsigned nM=m.size();
	for (unsigned j = 0; j < nM; ++j)
	{
		cout<<"|";
		cout<<m[j];
		cout<<"|"<<endl;
	}

	cout<<endl;
}
funcion generateF_INewton(lista coordenadas)
{
	funcion fun;
	matrix matriz(coordenadas.size(),row(coordenadas.size()+1));
	vector<number> coeficientes;

		matriz[0][0]=coordenadas[0]['x'];
		matriz[0][1]=coordenadas[0]['y'];
		coeficientes.push_back(coordenadas[0]['y']);
	for (number i = 1; i < matriz.size(); ++i)
	{	matriz[i][0]=coordenadas[i]['x'];
		matriz[i][1]=coordenadas[i]['y'];
		for (number j = 2; j < i+2; ++j)
		{
			matriz[i][j]=(matriz[i][j-1]-matriz[i-1][j-1])/(matriz[i][0]-matriz[i-j+1][0]);
			if (i==j-1)
			{
				coeficientes.push_back(matriz[i][j]);
			}
		}
	}
	print(coeficientes);
```

```
	// print(matriz);
	fun=make_pair(coeficientes,coordenadas);
	return fun;
}
number evaluar_funcion(funcion fun,number x)
{	number ans=0;
	ans+=fun.first[0];

	for (int i = 1; i < fun.first.size(); ++i)
	{
		number mult=1;
		mult*=fun.first[i];
		for (int j = 0; j < i; ++j)
		{
			mult*=(x-fun.second[j]['x']);
		}
		ans+=mult;
	}

	return ans;
}
void llenararchivo(funcion fun,number inicio,number final)
{
	ofstream archivo("interpolacionN.csv");
	for (number i = inicio; i < final; i+=0.5)
	{
		archivo<<i<<" "<<evaluar_funcion(fun,i)<<endl;
		// cout<<i<<" "<<evaluar_funcion(fun,i)<<endl;
	}
	archivo.close();
}
int main(int argc, char *argv[])
{

	// lista coordenadas;
	// llenarcoordenadas(coordenadas);
	lista coordenadas(40);
	llenarcoordenadas(coordenadas,0,20,0.5);
	funcion funcion1 =generateF_INewton(coordenadas);
	llenararchivo(funcion1,-2,22);

	return 0;
}
```