Temat: Interpolacja Newtona			
Wykonał: Marcin Fabrykowski	Wydział: FiIS	Kierunek Inf. Stos.	Grupa: grupa 3

1. Wstęp
   Aby omówić metodą interpolacji Newtona, niezbędne będzie poznanie pojęcia ilorazu różnicowego. Definiujemy go jako: $f(x_0; x_1) = \frac{f(x_1) - f(x_0)}{x_1 - x_0}$.
   Używając powyższego wzoru tworzymy tabelkę ilorazów. Wygląd tej tabelki pokaże w części *Wykonanie ćwiczenia*.
   Następnie nasz szukany wielomian interpolujący wyraża się wzorem:

   $$W_n(x) = f(x_0)+f(x_0;x_1)\omega_0(x)+f(x_0;x_1;x_2)\omega_1(x)+\cdots+f(x_0;x_1;\ldots;x_n)\omega_{n-1}(x)$$

2. wykonanie ćwiczenia
   Naszym celem będzie interpolacja funkcji $f(x) = \frac{1}{1+x^2}$. Zakładamy węzły interpolacji w punktach $x_i = -5, -2, -\frac{1}{2}, 0, \frac{1}{2}, 2, 5$.
   Obliczamy wartości funkcji w węzłach, a następnie wykonać tabelkę ilorazów różnicowych. Mając te wartości, obliczamy wielomian interpolujący.
   Powyższe zadania realizuje poniższy program.

Listing 1: main.cpp

```
#include <iostream>
#include <stdio.h>
using namespace std;
float f(float x);
float w(float x, float** tab, int k);
float W(float x,float **tab,int k);
void printTab(float** tab,int w,int h);
int main()
{
        int ileWezlow;
        cout<<"Podaj ilosc wezlow: ";
        cin>>ileWezlow;
        int i;
        float **tab=(float**)new float[ileWezlow+1];
```

```
        for(i=0;i<ileWezlow+1;i++)
        {
                tab[i]=new float[ileWezlow];
        };
        for(i=0;i<ileWezlow;i++)
        {
                cout<<"Podaj wezel nr: "<<i<<": ";
                cin>>tab[0][i];
        };
        for(i=0;i<ileWezlow;i++)
        {
                tab[1][i]=f(tab[0][i]);
        };
        int j;
        cout<<"\t"<<endl;
//      for(j=2;j<3;j++)
        for(j=2;j<ileWezlow+1;j++)
        {
                for(i=0;i<ileWezlow+1-j;i++)
                {
//
  tab[j][i]=(tab[i+1][j-1]-tab[i][j-1])/(tab[i+j-1][0]-tab[i][0]);
//
  tab[j][i]=(tab[j][i+1]-tab[j][i])/(tab[0][i+j+1]-tab[0][i]);
//
  cout<<tab[j-1][i+1]<<"\t"<<tab[j-1][i]<<endl;
//
  cout<<tab[0][i+j-1]<<"\t"<<tab[0][i]<<endl;
                        float
                            tmp1=tab[j-1][i+1]-tab[j-1][i];
                        float tmp2=tab[0][i+j-1]-tab[0][i];
//                      cout<<tmp1<<"\t"<<tmp2<<endl;
//                      cout<<"*****"<<endl;
                        tab[j][i]=tmp1/tmp2;
                };
        };
        printf("\n");
        printTab(tab,ileWezlow+1,ileWezlow);

        float m;
        FILE* plik;
        plik=fopen("w.dat","w");
        for(m=-5;m<5;m+=0.1)
        {
                fprintf(plik,"%f %f
                    %f\n",m,W(m,tab,ileWezlow),W(m,tab,1));
        };
        fclose(plik);
```

```
};
float f(float x)
{
        return 1/(1+x*x);
};
void printTab(float** tab,int w,int h)
{
        int x,y;
        for(x=0;x<h;x++)
        {
                for(y=0;y<w;y++)
                {
                        printf("%.3f\t",tab[y][x]);
                };
                printf("\n");
        };
};
float w(float x, float** tab, int k)
{
        float w=1;
        int i;
        for(i=0;i<k;i++)
        {
//                cout<<x<<"\t"<<tab[0][i]<<endl;
                w*=(x-tab[0][i]);
        };
        return w;
};

float W(float x,float** tab,int k)
{
        int i=0;
        float wynik=0;
        for(i=1;i<=k;i++)
        {
        //      cout<<w(x,tab,i-1)<<endl;
        //      cout<<tab[i][0]<<endl;
        //      cout<<"******"<<endl;

                wynik+=tab[i][0]*w(x,tab,i-1);
        };
        return wynik;
};
```

Czego wynikiem jest poniższa tabelka ilorazów różnicowych:

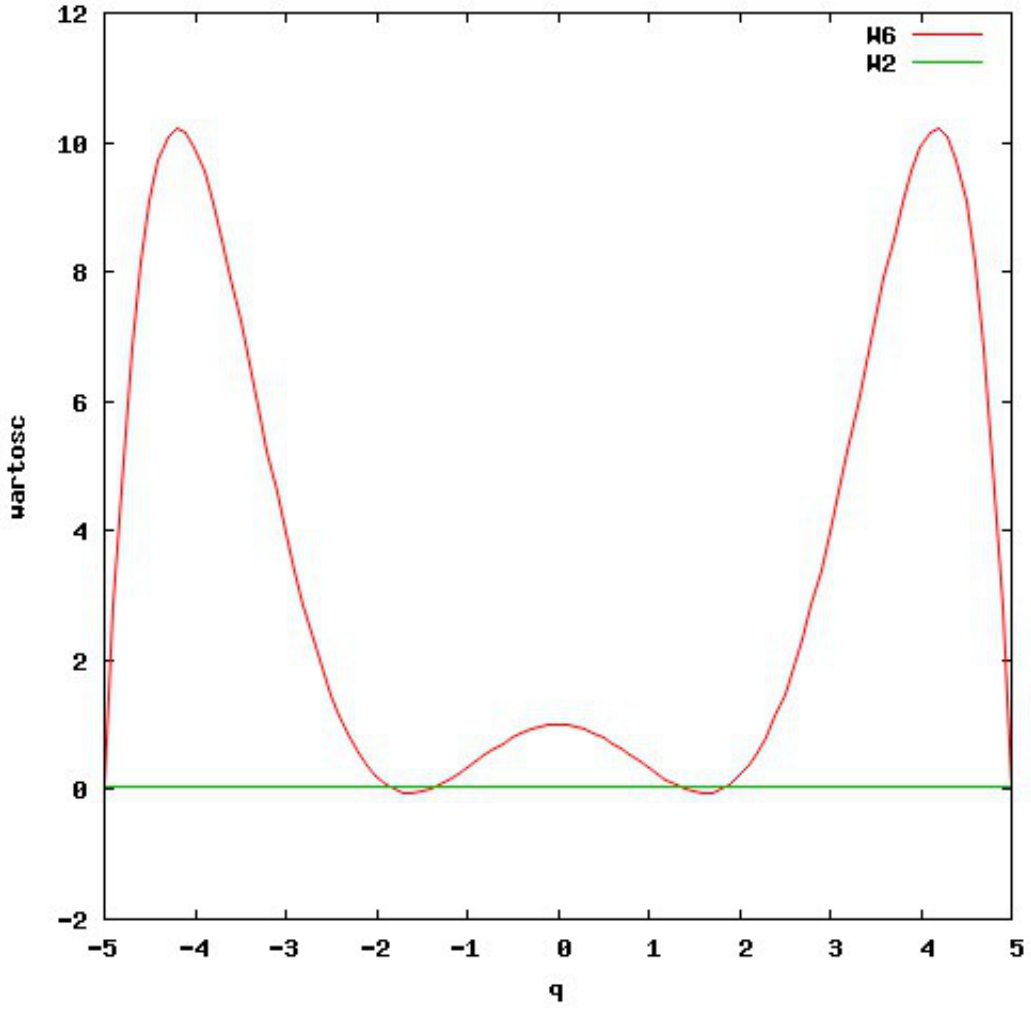
```
-5.000 0.038 0.054 0.077 -0.015 -0.055 0.031 -0.006
-2.000 0.200 0.400 -0.000 -0.320 0.160 -0.031 0.000
```

```
-0.500 0.800 0.400 -0.800 0.320 -0.055 0.000 0.000
0.000 1.000 -0.400 -0.000 0.015 0.000 0.000 0.000
0.500 0.800 -0.400 0.077 0.000 0.000 0.000 0.000
2.000 0.200 -0.054 0.000 0.000 0.000 0.000 0.000
5.000 0.038 0.000 0.000 0.000 0.000 0.000 0.000
```

Oraz interpolowane wartości funkcji:

```
-5.000000 0.038462 0.038462
-4.900000 2.820679 0.038462
-4.800000 5.057645 0.038462
-4.700000 6.811962 0.038462
-4.600000 8.141579 0.038462
-4.500000 9.099996 0.038462
-4.400001 9.736467 0.038462
-4.300001 10.096213 0.038462
-4.200001 10.220607 0.038462
-4.100001 10.147370 0.038462
-4.000001 9.910771 0.038462
-3.900001 9.541790 0.038462
-3.800001 9.068313 0.038462
-3.700001 8.515303 0.038462
-3.600001 7.904968 0.038462
-3.500001 7.256932 0.038462
-3.400002 6.588391 0.038462
-3.300002 5.914276 0.038462
-3.200002 5.247399 0.038462
-3.100002 4.598610 0.038462
-3.000002 3.976934 0.038462
-2.900002 3.389714 0.038462
-2.800002 2.842744 0.038462
-2.700002 2.340401 0.038462
-2.600002 1.885774 0.038462
-2.500002 1.480778 0.038462
-2.400002 1.126281 0.038462
-2.300003 0.822211 0.038462
-2.200003 0.567665 0.038462
-2.100003 0.361014 0.038462
-2.000003 0.200004 0.038462
-1.900003 0.081850 0.038462
-1.800003 0.003326 0.038462
-1.700003 -0.039147 0.038462
-1.600003 -0.049421 0.038462
-1.500003 -0.031539 0.038462
-1.400003 0.010330 0.038462
-1.300003 0.071969 0.038462
-1.200003 0.149170 0.038462
-1.100003 0.237797 0.038462
-1.000003 0.333844 0.038462
```

```
-0.900003 0.433478 0.038462
-0.800003 0.533094 0.038462
-0.700003 0.629354 0.038462
-0.600003 0.719221 0.038462
-0.500003 0.799998 0.038462
-0.400003 0.869354 0.038462
-0.300003 0.925348 0.038462
-0.200003 0.966450 0.038462
-0.100003 0.991557 0.038462
-0.000003 1.000000 0.038462
0.099997 0.991558 0.038462
0.199997 0.966452 0.038462
0.299997 0.925351 0.038462
0.399997 0.869357 0.038462
0.499997 0.800002 0.038462
0.599997 0.719225 0.038462
0.699997 0.629359 0.038462
0.799998 0.533099 0.038462
0.899998 0.433483 0.038462
0.999998 0.333849 0.038462
1.099998 0.237802 0.038462
1.199998 0.149174 0.038462
1.299998 0.071972 0.038462
1.399998 0.010332 0.038462
1.499998 -0.031538 0.038462
1.599998 -0.049420 0.038462
1.699998 -0.039149 0.038462
1.799998 0.003323 0.038462
1.899998 0.081845 0.038462
1.999998 0.199996 0.038462
2.099998 0.361004 0.038462
2.199998 0.567653 0.038462
2.299998 0.822197 0.038462
2.399997 1.126265 0.038462
2.499997 1.480758 0.038462
2.599997 1.885753 0.038462
2.699997 2.340377 0.038462
2.799997 2.842716 0.038462
2.899997 3.389685 0.038462
2.999997 3.976903 0.038462
3.099997 4.598579 0.038462
3.199997 5.247366 0.038462
3.299997 5.914240 0.038462
3.399997 6.588358 0.038462
3.499996 7.256896 0.038462
3.599996 7.904938 0.038462
3.699996 8.515272 0.038462
3.799996 9.068290 0.038462
3.899996 9.541770 0.038462
```




```
3.999996 9.910754 0.038462
4.099996 10.147364 0.038462
4.199996 10.220609 0.038462
4.299996 10.096224 0.038462
4.399996 9.736492 0.038462
4.499996 9.100030 0.038462
4.599996 8.141636 0.038462
4.699996 6.812039 0.038462
4.799995 5.057743 0.038462
4.899995 2.820802 0.038462
4.999995 0.038606 0.038462
```

Co przedstawiając graficznie jako funkcję interpolowaną i interpolującą widać na rys.

3. Wnioski
   Interpolacja Newtona pozwala na wyliczenie wartości funkcji w argumentach nie będących węzłami interpolacyjnymi. Jednakże wykonanie ćwiczenia nie dało oczekiwanego wyniku interpolacyjnego, co mogło być spowodowane błędem implementacyjnym w zastosowanym programie autorskim.