Схема линейной интерполяции:

y=a0 +a1x

a1=(y1 - y0)/(x1 - x0)

a0=y0 - a1x0

**//Интерполяция линейным полиномом**

// y=a0+a1\*x

//a1=(y1-y0)/(x1-x0)

//a0=y0-a1\*x0

//Драйвер линейной интерполяции

#include <stdio.h>

#include <stdlib.h>

#include<math.h>

float f(float x);

float li(float x0,float y0,float x1,float y1,float v[],int n);

int main()

{

const int n=10;

float v[n];

float x0,y0,x1,y1,x2,y2,r;

x0=0;

y0=0;

x1=1;

y1=-1;

r=li(x0,y0,x1,y1,v,n);

system("pause");

return 0;

}

//Функция вычисления значений

float f(float x)

{return x\*x-2\*x;}

//Функция линейной интерполяции

float li(float x0,float y0,float x1,float y1,float v[],int n)

{

float a1=(y1-y0)/(x1-x0);

float a0=y0-a1\*x0;

float x=x0;

float h=(x1-x0)/n;

v[0]=x0;

for(int i=1; i<=n+1; i++)

{

v[i]=a0+a1\*x;

printf(" %f %f %f \n",x,v[i],f(x));

x+=h;

}

return 0;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Схема квадратичной интерполяции

y= a0 + a1x + a2x2

a2 = (y2 - 2y1 + y0)/2h2

a1 =(y1 - y0)/h - a2(x1 + x0)

a0 = y0 - a1x0 - a2x02

**//Интерполяция квадратичным полиномом**

// y=a0+a1\*x+a2\*x2

//a2=(y2-2y1+y0)/(x1-x0)2/2

//a1=(y1-y0)/(x1-x0)-a2(x1+x2)

//a0=y0-a1\*x0=a2\*x02

////Драйвер квадратичной интерполяции

#include <stdio.h>

#include <stdlib.h>

#include<math.h>

float f(float x);

float qwi(float x0,float y0,float x1,float y1,float x2,float y2,float v[],int n);

int main()

{

const int n=10;

float v[n];

float x0,y0,x1,y1,x2,y2,r;

x0=0.;

y0=0.;

x1=1.;

y1=-1.;

x2=2;

y2=0;

r=qwi(x0,y0,x1,y1,x2,y2,v,n);

system("pause");

return 0;

}

//Функция вычисления значений

float f(float x)

{return x\*x-2\*x;}

//Функция квадратичной интерполяции

float qwi(float x0,float y0,float x1,float y1,float x2,float y2,float v[],int n)

{

float H=x1-x0;

float a2=(y2-2\*y1+y0)/2/H/H;

float a1=(y1-y0)/H-a2\*(x1+x0);

float a0=y0-a1\*x0-a2\*x0\*x0;

float x=x0;

float h=H/n;

v[0]=x0;

for(int i=1; i<=n+1; i++)

{

v[i]=a0+a1\*x+a2\*x\*x;

printf(" %f %f %f \n",x,v[i],f(x));

x+=h;

}

return 0;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Схема линейной интерполяции Ньютона**

Pn(x) = С0 + С1(x - x0) + C2(x - x0) (x - x1)





**//Интерполяция линейным полиномом Ньютона**

// y=c0+c1\*(x-x0)

//c0=y0

//c1=(y1-y0)/(x1-x0)

//Драйвер линейной интерполяции Ньютона

#include <stdio.h>

#include <stdlib.h>

#include<math.h>

float f(float x);

float lin(float x0,float y0,float x1,float y1,float v[],int n);

int main()

{

const int n=10;

float v[n];

float x0,y0,x1,y1,x2,y2,r;

x0=0;

y0=0;

x1=1;

y1=-1;

r=lin(x0,y0,x1,y1,v,n);

system("pause");

return 0;

}

//Функция вычисления значений

float f(float x)

{return x\*x-2\*x;}

//Функция линейной интерполяции Ньютона

float lin(float x0,float y0,float x1,float y1,float v[],int n)

{

float H=x1-x0;

float c0=y0;

float c1=(y1-y0)/H;

float x=x0;

float h=(x1-x0)/n;

v[0]=x0;

for(int i=1; i<=n+1; i++)

{

v[i]=c0+c1\*(x-x0);

printf(" %f %f %f \n",x,v[i],f(x));

x+=h;

}

return 0;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Схемаквадратичной интерполяции Ньютона**

Pn(x) = С0 + С1(x - x0) + C2(x - x0) (x - x1)







**//Интерполяция квадратичным полиномом Ньютона**

// y=c0+c1\*(x-x0)

//c0=y0;

//c1=(y1-y0)/(x1-x0)

//c2=((y2-y0)/x2-x0)-(y1-y0)/(x1-x0))/(x2-x0)

//Драйвер квадратичного полинома Ньютона

#include <stdio.h>

#include <stdlib.h>

#include<math.h>

float f(float x);

float qwn(float x0,float y0,float x1,float y1,float x2,float e2,float v[],int n);

int main()

{

const int n=10;

float v[n];

float x0,y0,x1,y1,x2,y2,r;

x0=0;

y0=0;

x1=1;

y1=-1;

x2=2;

y2=0;

r=qwn(x0,y0,x1,y1,x2,y2,v,n);

system("pause");

return 0;

}

//Функция вычисления значений

float f(float x)

{return x\*x-2\*x;}

//Функция квадратичной интерполяции Ньютона

float qwn(float x0,float y0,float x1,float y1,float x2,float y2,float v[],int n)

{

float H=x1-x0;

float c0=y0;

float c1=(y1-y0)/H;

float c2=(y2-2.\*y1+y0)/4/H/H;

float x=x0;

float h=(x1-x0)/n;

v[0]=x0;

for(int i=1; i<=n+1; i++)

{

v[i]=c0+c1\*(x-x0)+c2\*(x-x0)\*(x-x1);

printf(" %f %f %f \n",x,v[i],f(x));

x+=h;

}

return 0;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Схема линейной интерполяции Лагранжа

**//Интерполяция линейным полиномом Лагранжа**

// y=y0\*(x-x1)/(x0-x1)+y1\*(x-x0)/(x1-x0)

//Драйвер линейной интерполяции Лагранжа

#include <stdio.h>

#include <stdlib.h>

#include<math.h>

float f(float x);

float lil(float x0,float y0,float x1,float y1,float v[],int n);

int main()

{

const int n=10;

float v[n];

float x0,y0,x1,y1,x2,y2,r;

x0=0;

y0=0;

x1=1;

y1=-1;

r=lil(x0,y0,x1,y1,v,n);

system("pause");

return 0;

}

//Функция вычисления значений

float f(float x)

{return x\*x-2\*x;}

//Функция линейной интерполяции Дагранжа

float lil(float x0,float y0,float x1,float y1,float v[],int n)

{

float H=x1-x0;

float x=x0;

float h=(x1-x0)/n;

v[0]=x0;

for(int i=1; i<=n+1; i++)

{

v[i]=-y0\*(x-x1)/H+y1\*(x-x0)/H;

printf(" %f %f %f \n",x,v[i],f(x));

x+=h;

}

return 0;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**//Интерполяция квадратичным полиномом Лагранжа**



**//Интерполяция квадратичным полиномом Лагранжа**

// y=y0\*(x-x1)\*(x-x2)/(x0-x1)/(x0-x2)+y1\*(x-x0)\*(x-x2)/(x1-x0)/(x1-x2)

// +y2\*(x-x0)\*(x-x1)/(x2-x0)/(x2-x1)

//Драйвер квадратичной интерполяции Лагранжа

#include <stdio.h>

#include <stdlib.h>

#include<math.h>

float f(float x);

float qwl(float x0,float y0,float x1,float y1,float x2,float e2,float v[],int n);

int main()

{

const int n=10;

float v[n];

float x0,y0,x1,y1,x2,y2,r;

x0=0;

y0=0;

x1=1;

y1=-1;

x2=2;

y2=0;

r=qwl(x0,y0,x1,y1,x2,y2,v,n);

system("pause");

return 0;

}

//Функция вычисления значений

float f(float x)

{return x\*x-2\*x;}

//Функция квадратичной интерполяции Лагранжа

float qwl(float x0,float y0,float x1,float y1,float x2,float y2,float v[],int n)

{

float H=x1-x0;

float x=x0;

float h=(x1-x0)/n;

v[0]=x0;

for(int i=1; i<=n+1; i++)

{

v[i]= y0\*(x-x1)\*(x-x2)/H/H/2.-y1\*(x-x0)\*(x-x2)/H/H+y2\*(x-x0)\*(x-x1)/H/H/2.;

printf(" %f %f %f \n",x,v[i],f(x));

x+=h;

}

return 0;

}