Санкт-Петербургский Государственный Политехнический Университет
Факультет Технической Кибернетики
Кафедра Информационных и Управляющих Систем

ОТЧЕТ
о лабораторной работе № 1

«Интерполяция»
по численному анализу

Работу выполнил студент:

Голубева А. С.
гр. 3084/1

Преподаватель:

Земницкий В. А.

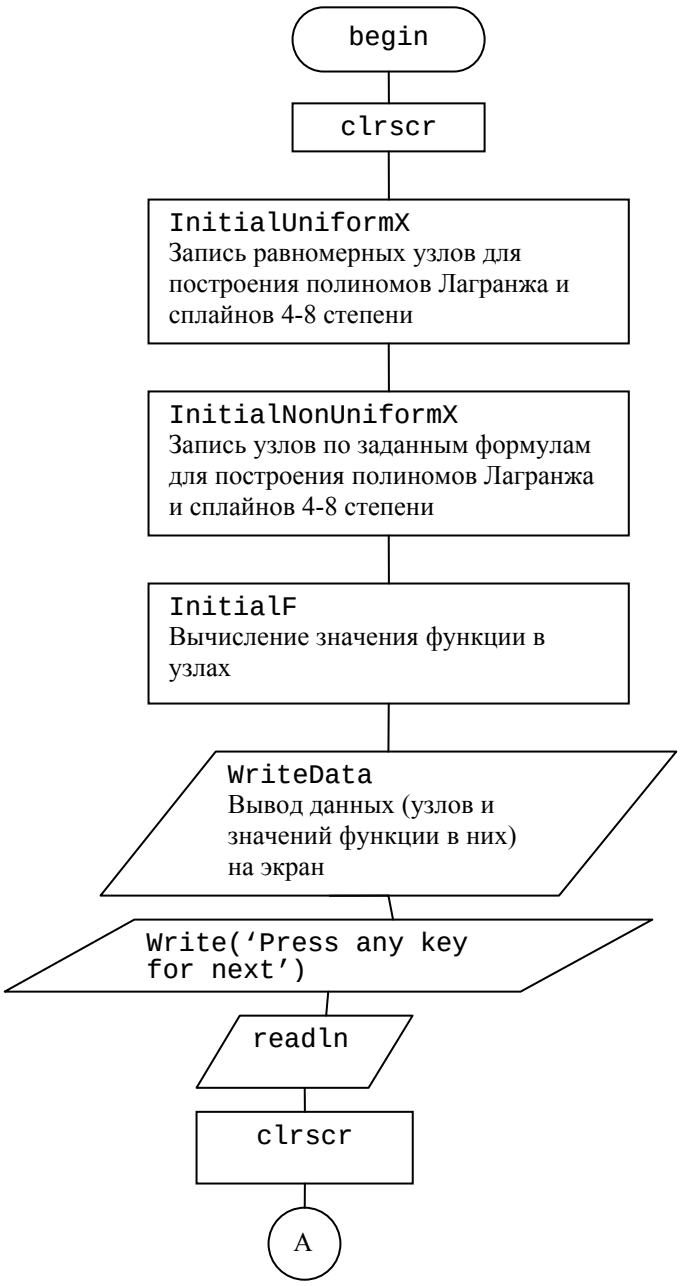
Санкт-Петербург
2004

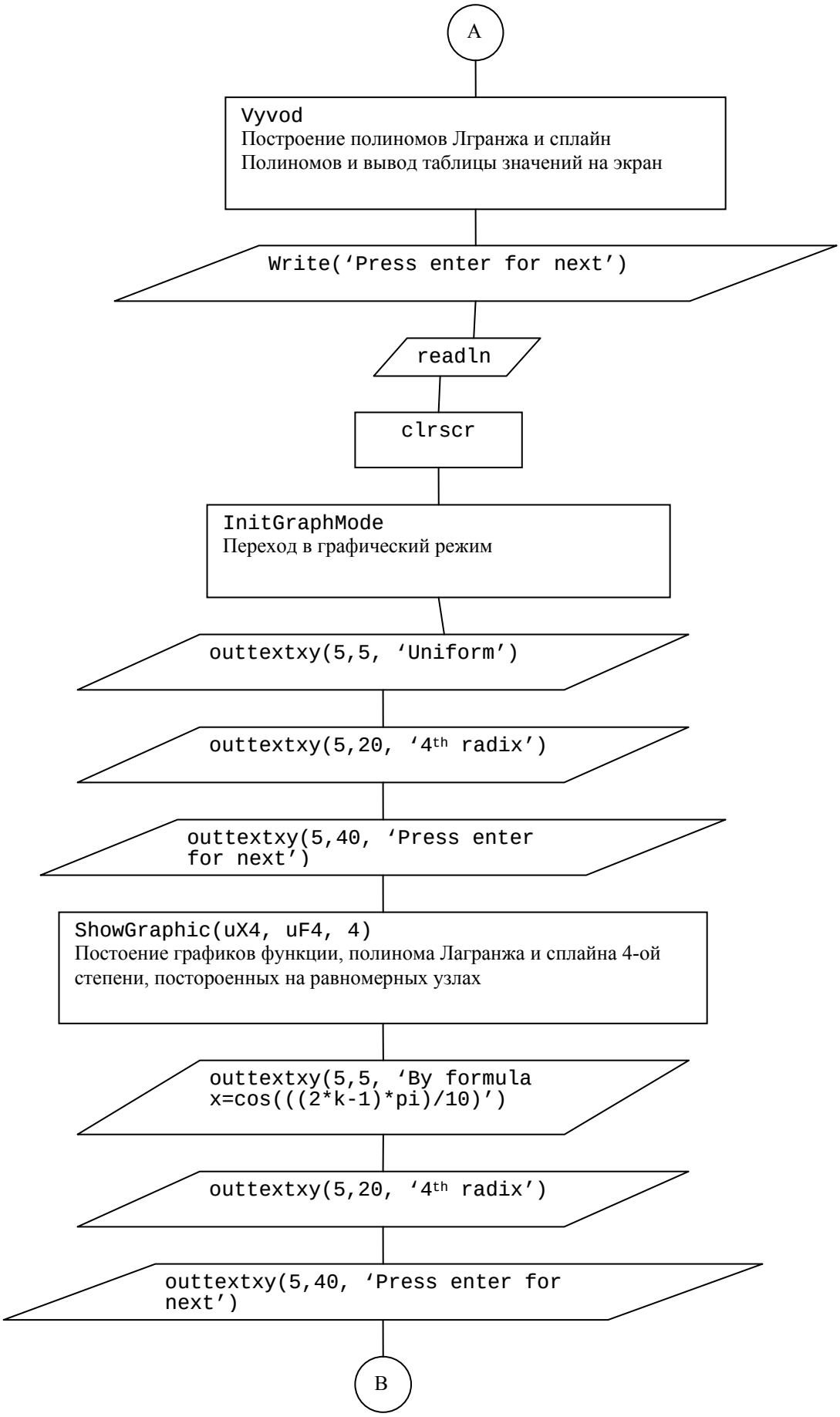
Постановка задачи.

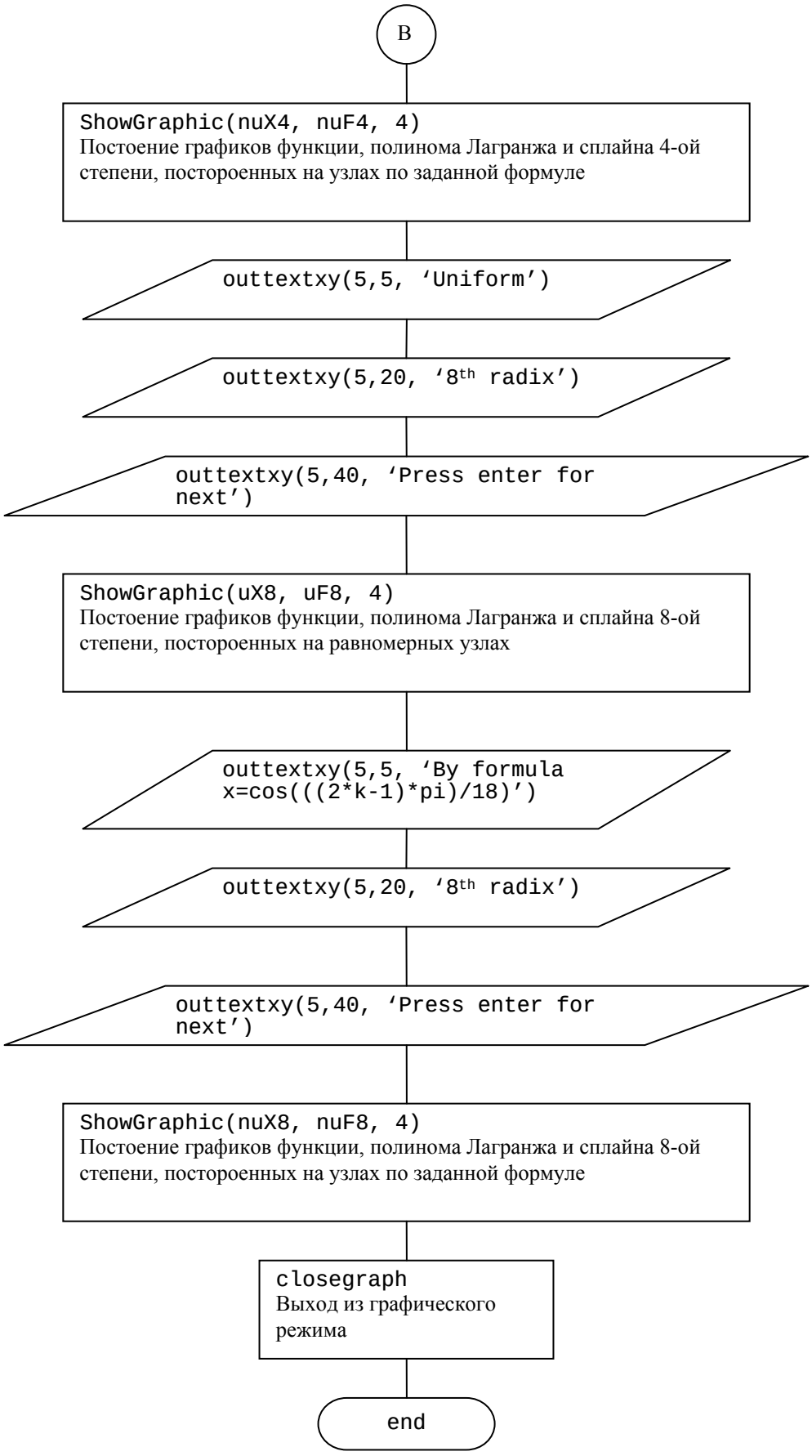
Для функции $f(x)=\frac{1}{1+25x^2}$, $x\in[-1,+1]$ построить интерполяционные полиномы Лагранжа 4-ой и 8-ой степени и соответствующие им сплайн-функции S(x) для двух наборов узлов. В первом узлы расположить равномерно, а во втором по формулам: $x_k=\cos\frac{2k-1}{10}\pi$, k = 5, 4, 3, 2, 1 (для полинома 4-ой степени) и $x_k=\cos\frac{2k-1}{18}\pi$, k= 9, 8, 7, 6, 5, 4, 3, 2, 1 (для полинома 8-ой степени).

Вычислительный эксперимент состоит в выявлении качественных и количественных характеристик интерполяционных кривых. Построить графики и проанализировать результаты.

1. Блок-схема алгоритма программы, строящей полиномы Лагранжа и сплайн-функции.




A
Vyvod
Построение полиномов Лгранжа и сплайн
Полиномов и вывод таблицы значений на экран
Write('Press enter for next')
readln
clrscr
InitGraphMode
Переход в графический режим
outtextxy(5,5, 'Uniform')
outtextxy(5,20, '4th radix')
outtextxy(5,40, 'Press enter
for next')
ShowGraphic(uX4, uF4, 4)
Постоение графиков функции, полинома Лагранжа и сплайна 4-ой
степени, постороенных на равномерных узлах
outtextxy(5,5, 'By formula
x=cos(((2*k-1)*pi)/10)')
outtextxy(5,20, '4th radix')
outtextxy(5,40, 'Press enter for
next')
B


B
ShowGraphic(nuX4, nuF4, 4)
Постоение графиков функции, полинома Лагранжа и сплайна 4-ой степени, постороенных на узлах по заданной формуле
outtextxy(5,5, ‘Uniform’)
outtextxy(5,20, ‘8th radix’)
outtextxy(5,40, ‘Press enter for next’)
ShowGraphic(uX8, uF8, 4)
Постоение графиков функции, полинома Лагранжа и сплайна 8-ой степени, постороенных на равномерных узлах
outtextxy(5,5, ‘By formula x=cos(((2*k-1)*pi)/18)’)
outtextxy(5,20, ‘8th radix’)
outtextxy(5,40, ‘Press enter for next’)
ShowGraphic(nuX8, nuF8, 4)
Постоение графиков функции, полинома Лагранжа и сплайна 8-ой степени, постороенных на узлах по заданной формуле
closegraph
Выход из графического режима
end

- Вычисление полинома Лагранжа реализовано по формуле:

$$L_n(x) = \sum_{i=o}^{n} f(x_i) \cdot \frac{(x-x_0)\ldots(x-x_{i-1})(x-x_{i+1})\ldots(x-x_n)}{(x_i-x_0)\ldots(x_i-x_{i-1})(x_i-x_{i+1})\ldots(x_i-x_n)}$$

- Вычисление сплайн-функции реализовано по примеру процедур SPLINE и SEVAL из библиотеки FMM пакета FORSYTHE

2. Исходный текст программы на Паскале, реализующей данный алгоритм (1.pas).

```
program LAB1;
uses crt, graph;
var

uX4, nuX4  : array[0..4] of real; {Uniform and non-uniform X for 4th radix}
uX8, nuX8  : array[0..8] of real; {Uniform and non-uniform X for 8th radix}

uF4, nuF4  : array[0..4] of real; {Uniform and non-uniform X for 4th radix}
uF8, nuF8  : array[0..8] of real; {Uniform and non-uniform X for 8th radix}

mB, mC, mD : array[0..8] of real; {Memory for spline procedure}
a, l, s : real;

{                 Initial uniform axis of ordinates
  x      - axis of ordinates
  dStep  - step size of uniform distribution
  nCount - number of items
}
Procedure InitialUniformX( var x: array of real; dStep : real; nCount : Integer);
  var i : Integer;
begin
  x[0] := -1;
  for i := 1 to nCount do x[ i] := x[ i - 1] + dStep;
end;

{                 Initial non-uniform axis of ordinates
  x      - axis of ordinates
  dParam - parameter of non-uniform distribution
  nCount - number of items
}
Procedure InitialNonUniformX( var x: array of real; dParam : real; nCount :
Integer);
  var k : real;
      i : Integer;
begin
  k := dParam-nCount;
  for i := 0 to nCount do
  begin
    k   := k - 1;
    x[i]:= cos( ( ( 2 * k - 1) * pi) / dParam);
  end;
end;

{                 Initial values of function for uniform and non-uniform
distribution
  uX     - axis of ordinates, uniform distribution
  nuX    - axis of ordinates, non-uniform distribution
  uF     - axis of abscissas for uniform distribution
  nuF    - axis of abscissas for non-uniform distribution
  nCount - number of items
}
Procedure InitialF( var uF, nuF: array of real; uX, nuX : array of real; nCount :
Integer);
  var i : Integer;
begin
  for i := 0 to nCount do
  begin
    uF[ i]  := 1 / ( 1 + 25 * sqr( uX[ i]));
    nuF[ i] := 1 / ( 1 + 25 * sqr( nuX[ i]));
```

```
  end;
end;

{                 Output data to display
  str    - legend of data
  X      - axis of ordinates (argument)
  F      - axis of abscissas (value of function)
  nCount - number of items
}
Procedure WriteData( str : string; X, F: array of real; nCount : Integer);
  var i : Integer;
begin
  writeln( str); writeln;
  for i := 0 to nCount do write( X[ i] : 8 : 3); writeln;
  for i := 0 to nCount do write( F[ i] : 8 : 3); writeln;
  writeln;
end;

{                 Interpolation by polynom Lagrange
  x      - axis of ordinates
  y      - axis of abscissas
  nCount - number of items
  xt     - point of axis of ordinates
  return - point of axis of abscissas
}
Function PolynomLagrange( x, y : array of real; nCount : integer; xt : real) : real;
var  i, j : integer;
     p, s : real;
begin
  s  := 0;
  for i := 0 to nCount do
  begin
    p := y[ i];
    for j := 0 to nCount do if( j <> i) then p := p * ( xt - x[ j]) / ( x[ i] - x[
j]);
    s  := s + p;
  end;

  PolynomLagrange := s;
end;

procedure GetBCD( x, y : array of real; var b, c, d : array of real;
nCount:integer);
 label 10,20,30;
 var   i, ib : integer;
       t     : real;

begin
  if nCount + 1 < 2 then exit;
  if nCount + 1 < 3 then goto 20;

  d[ 0] := x[ 1] - x[ 0];
  c[ 1] := ( y[ 1] - y[ 0]) / d[ 0];
  for i := 1 to nCount - 1 do
  begin
    d[ i]     := x[ i + 1] - x[ i];
    b[ i]     := 2 * ( d[ i - 1] + d[ i]);
    c[ i + 1] := ( y[ i + 1] - y[ i]) / d[ i];
    c[ i]     := c[ i + 1] - c[ i];
  end;
  b[ 0]      := -d[ 0];
  b[ nCount] := -d[ nCount - 1];
  c[ 0]      :=  0;
  c[ nCount] :=  0;

  if nCount + 1 = 3 then goto 10;
  c[ 0]      :=  c[ 2] / (x[ 3] - x[ 1]) - c[ 1] / ( x[ 2] - x[ 0]);
  c[ nCount] :=  c[ nCount - 1] / ( x[ nCount] - x[ nCount - 2]) - c[ nCount - 2] /
( x[ nCount - 1] - x[ nCount - 3]);
  c[ 0]      :=  c[ 0] * sqr( d[ 0]) / ( x[ 3] - x[ 0]);
  c[ nCount] := -c[ nCount] * sqr( d[ nCount - 1]) / ( x[ nCount] - x[ nCount - 3]);
```

```
10:
  for i := 1 to nCount do
  begin
    t     := d[ i - 1] / b[ i - 1];
    b[ i] := b[ i] - t * d[ i - 1];
    c[ i] := c[ i] - t * c[ i - 1];
  end;

  c[ nCount] := c[ nCount] / b[ nCount];
  for ib := 0 to nCount - 1 do
  begin
    i     := nCount - ib - 1;
    c[ i] := ( c[ i] - d[ i] * c[ i + 1]) / b[ i];
  end;

  b[ nCount] := ( y[ nCount] - y[ nCount - 1]) / d[ nCount - 1] + d[ nCount - 1] * (
c[ nCount - 1] + 2 * c[ nCount]);
  for i := 0 to nCount - 1 do
  begin
    b[ i] := (y[ i + 1] - y[ i]) / d[ i] - d[ i] * ( c[ i + 1] + 2 * c[ i]);
    d[ i] := (c[ i + 1] - c[ i]) / d[ i];
    c[ i] := 3 * c[ i];
  end;

  c[ nCount] := 3 * c[ nCount];
  d[ nCount] := d[ nCount - 1];
  exit;

20:
  b[ 0] := ( y[ 1] - y[ 0]) / ( x[ 1] - x[ 0]);
  c[ 0] := 0;
  d[ 0] := 0;
  b[ 1] := b[ 0];
  c[ 1] := 0;
  d[ 1] := 0;

30:
end;

function Seval( xt : real; x, y, b, c, d : array of real; nCount : integer):real;
  label 10,20,30;
  var i, j, k:integer;
      dx     :real;
begin
  i := 0;
  if i >= nCount     then i := 0;
  if xt < x[ i]      then goto 10;
  if xt <= x[ i + 1] then goto 30;

10:
  i := 0;
  j := nCount + 1;

20:
  k := ( i + j) div 2;
  if xt <  x[ k]  then j := k;
  if xt >= x[ k]  then i := k;
  if j > ( i + 1) then goto 20;

30:
  dx    := xt - x[ i];
  Seval := y[ i] + dx * ( b[ i] + dx * ( c[ i] + dx * d[ i]));
end;

{                 Interpolation by spline
  x      - axis of ordinates
  y      - axis of abscissas
  nCount - number of items
  xt     - point of axis of ordinates
  return - point of axis of abscissas
}
function Spline( x, y : array of real; nCount : integer; xt : real) : real;
```

```
begin
  GetBCD( x, y, mB, mC, mD, nCount);
  Spline := Seval( xt, x, y, mB, mC, mD, nCount);
end;

Procedure Vyvod(x, y : array of real; nCount: integer; str: string);
var xt, s, l: real;
begin
  xt:=-1.1;
    WriteLn(str);
    WriteLn('x':12, 'Lagrange':12, 'Spline':12);
  repeat
  xt := xt + 0.1;
  l := PolynomLagrange(x, y, nCount, xt);
  s := Spline(x, y, nCount, xt);
  Writeln(xt:12:6, l:12:6, s:12:6);
  until xt > 1;
end;

{                Initialize graphic mode
}
Procedure InitGraphMode;
  var driver, mode, error: integer;
begin
  driver := detect;                                    {Detect device driver}
  initgraph(driver, mode, 'e:\bp70\bp\bgi'{'C:\'});  {Initialize graphic mode}
  if error<>grOk then writeln(grapherrormsg(error));  {If initialize is not done
then put error message}
end;


{                Output data to display (video mode)
  Show function, Lagrange intepolation, spline interpolation
  x      - axis of ordinates (argument)
  y      - axis of abscissas (value of function)
  nCount - number of items
}
Procedure ShowGraphic( x, y : array of real; nCount: integer);
var
  xt, yt: real;
  s     : string;
begin

  {Show axis}
  line(319, 1, 319, 470);
  line(1, 240, 639, 240);

  {Show points inside axis of ordinates}
  xt := -1.5;
  repeat
    if( ( xt > -0.1) and ( xt < 0.1)) then  xt := xt + 0.2;
    str( xt:3:1, s); outtextxy( round( 320 + 150 * xt - 5), 250, s);
    line( round( 320 + 150 * xt), 235, round( 320 + 150 * xt), 245);
    xt := xt + 0.2;
  until xt > 1.5;

  {Show points inside axis of abscissas}
  yt:=-1.1;
  repeat
    str( yt:3:1, s); outtextxy( 330, round( 240 - 230 * yt - 3), s);
    line( 315, round( 240 - 230 * yt), 325, round( 240 - 230 * yt));
    if yt = -0.1 then yt := 0;
    yt := yt + 0.2;
  until yt > 1;

  {Show legend of graphic}
  setcolor(3);
  line(450, 13, 480, 13);
  setcolor(5);
  line(450, 23, 480, 23);
  setcolor(2);
  line(450, 33, 480, 33);
```

```
  setcolor(15);
  outtextxy(490, 10, 'Function');
  outtextxy(490, 20, 'Polynome Lagrange');
  outtextxy(490, 30, 'Spline-polynome');

  {Show graphics}
  xt := -1;
  repeat
    {Show point of function}
    yt := 1 / ( 1 + 25 * sqr( xt));
    putpixel( round( 320 + 150 * xt), round( 240 - 230 * yt), 3);

    {Show point of function by Lagrange interpolation}
    yt := PolynomLagrange( x, y, nCount, xt);
    putpixel( round( 320 + 150 * xt), round( 240 - 230 * yt), 5);

    {Show point of function by spline interpolation}
    yt := Spline( x, y, nCount, xt);
    putpixel( round( 320 + 150 * xt), round( 240 - 230 * yt), 2);

    {Next point}
    xt := xt + 0.005;
  until xt > 1;

  {Press enter key and clear graphic device}
  Readln;
  cleardevice;
end;

{Start program}
begin
  clrscr;  {Clear screen}

  {Initialize of data}
  InitialUniformX( uX4, 0.5,  4);
  InitialUniformX( uX8, 0.25, 8);

  InitialNonUniformX( nuX4, 10, 4);
  InitialNonUniformX( nuX8, 18, 8);

  InitialF( uF4, nuF4, uX4, nuX4, 4);
  InitialF( uF8, nuF8, uX8, nuX8, 8);

  {Output data to table of screen}
  WriteData( 'Uniform 4th radix', uX4, uF4, 4);
  WriteData( 'By formula x=cos(((2*k-1)*pi)/10) 4th radix', nuX4, nuF4, 4);
  WriteData( 'Uniform 8th radix', uX8, uF8, 8);
  WriteData( 'By formula x=cos(((2*k-1)*pi)/18) 8th radix', nuX8, nuF8, 8);

  {Any key pressed pliz}
  Write( ' Press any key for next...'); repeat until KeyPressed;
  readln;

  clrscr;
  Vyvod(uX4, uF4, 4, 'Uniform 4th radix');
  Write( ' Press enter for next...'); repeat until KeyPressed;
  readln;

  clrscr;
  Vyvod(nuX4, nuF4, 4, 'By formula x=cos(((2*k-1)*pi)/10) 4th radix');
  Write( ' Press enter for next...'); repeat until KeyPressed;
  readln;

  clrscr;
  Vyvod(uX8, uF8, 8, 'Uniform 8th radix');
  Write( ' Press enter for next...'); repeat until KeyPressed;
  readln;

  clrscr;
  Vyvod(nuX8, nuF8, 8, 'By formula x=cos(((2*k-1)*pi)/18) 8th radix');
  Write( ' Press enter for next...'); repeat until KeyPressed;
  readln;
```

```
  {Initialize graphic mode}
  InitGraphMode;

  {Show graphic for uniform distribution of X and 4th radix}
  outtextxy(5,5, 'Uniform');
  outtextxy(5,20,'4th radix');
  outtextxy(5,40,'Press enter for next...');
  ShowGraphic(uX4, uF4, 4);

  {Show graphic for non-uniform distribution of X and 4th radix}
  outtextxy(5,5, 'By formula x=cos(((2*k-1)*pi)/10)');
  outtextxy(5,20,'4th radix');
  outtextxy(5,40,'Press enter for next...');
  ShowGraphic(nuX4, nuF4, 4);

  {Show graphic for uniform distribution of X and 8th radix}
  outtextxy(5,5, 'Uniform');
  outtextxy(5,20,'8th radix');
  outtextxy(5,40,'Press enter for next...');
  ShowGraphic(uX8, uF8, 8);

  {Show graphic for non-uniform distribution of X and 8th radix}
  outtextxy(5,5, 'By formula x=cos(((2*k-1)*pi)/18)');
  outtextxy(5,20,'8th radix');
  outtextxy(5,40,'Press enter for next...');
  ShowGraphic(nuX8, nuF8, 8);

  {Close graphic mode}
  closegraph;
end.
```

3. Результаты работы программы (1.exe)

- Таблица узлов и их значений:

```
Uniform 4th radix

-1.000  -0.500   0.000   0.500   1.000
 0.038   0.138   1.000   0.138   0.038

By formula x=cos(((2*K-1)*pi)/10) 4th radix

-0.951  -0.588   0.000   0.588   0.951
 0.042   0.104   1.000   0.104   0.042

Uniform 8th radix

-1.000  -0.750  -0.500  -0.250   0.000   0.250   0.500   0.750
 0.038   0.066   0.138   0.390   1.000   0.390   0.138   0.066

By formula x=cos(((2*K-1)*pi)/18) 8th radix

-0.985  -0.866  -0.643  -0.342   0.000   0.342   0.643   0.866
 0.040   0.051   0.088   0.255   1.000   0.255   0.088   0.051

Press any key for next..._
```

- Таблицы значений полиномов Лагранжа и сплайн полиномов в точках промежутка $x \in [-1,+1]$:

Uniform 4th radix

x	Lagrange	Spline
-1.000000	0.038462	0.038462
-0.900000	-0.289125	-0.144120
-0.800000	-0.379310	-0.205570
-0.700000	-0.299735	-0.165782
-0.600000	-0.110080	-0.044651
-0.500000	0.137931	0.137931
-0.400000	0.400531	0.359859
-0.300000	0.641910	0.590186
-0.200000	0.834218	0.795756
-0.100000	0.957560	0.943413
0.000000	1.000000	1.000000
0.100000	0.957560	0.943413
0.200000	0.834218	0.795756
0.300000	0.641910	0.590186
0.400000	0.400531	0.359859
0.500000	0.137931	0.137931
0.600000	-0.110080	-0.044651
0.700000	-0.299735	-0.165782
0.800000	-0.379310	-0.205570
0.900000	-0.289125	-0.144120
1.000000	0.038462	0.038462

Press enter for next...

By formula x=cos(((2*K-1)*pi)/10) 4th radix

x	Lagrange	Spline
-1.000000	0.203516	0.129387
-0.900000	-0.067839	-0.020161
-0.800000	-0.142543	-0.066806
-0.700000	-0.076627	-0.026221
-0.600000	0.080472	0.085921
-0.500000	0.285910	0.252671
-0.400000	0.503433	0.449674
-0.300000	0.703378	0.649922
-0.200000	0.862675	0.826401
-0.100000	0.964845	0.952098
0.000000	1.000000	1.000000
0.100000	0.964845	0.952098
0.200000	0.862675	0.826401
0.300000	0.703378	0.649922
0.400000	0.503433	0.449674
0.500000	0.285910	0.252671
0.600000	0.080472	0.085921
0.700000	-0.076627	-0.026221
0.800000	-0.142543	-0.066806
0.900000	-0.067839	-0.020161
1.000000	0.203516	0.129387

Press enter for next...

Uniform 8th radix

x	Lagrange	Spline
-1.000000	0.038462	0.038462
-0.900000	-0.960063	0.047948
-0.800000	-0.258703	0.058218
-0.700000	0.263777	0.077702
-0.600000	0.304934	0.106809
-0.500000	0.137931	0.137931
-0.400000	0.072570	0.177056
-0.300000	0.237372	0.285961
-0.200000	0.563623	0.534059
-0.100000	0.874004	0.845666
0.000000	1.000000	1.000000
0.100000	0.874004	0.845666
0.200000	0.563623	0.534059
0.300000	0.237372	0.285961
0.400000	0.072570	0.177056
0.500000	0.137931	0.137931
0.600000	0.304934	0.106809
0.700000	0.263777	0.077702
0.800000	-0.258703	0.058218
0.900000	-0.960063	0.047948
1.000000	0.038462	0.038462

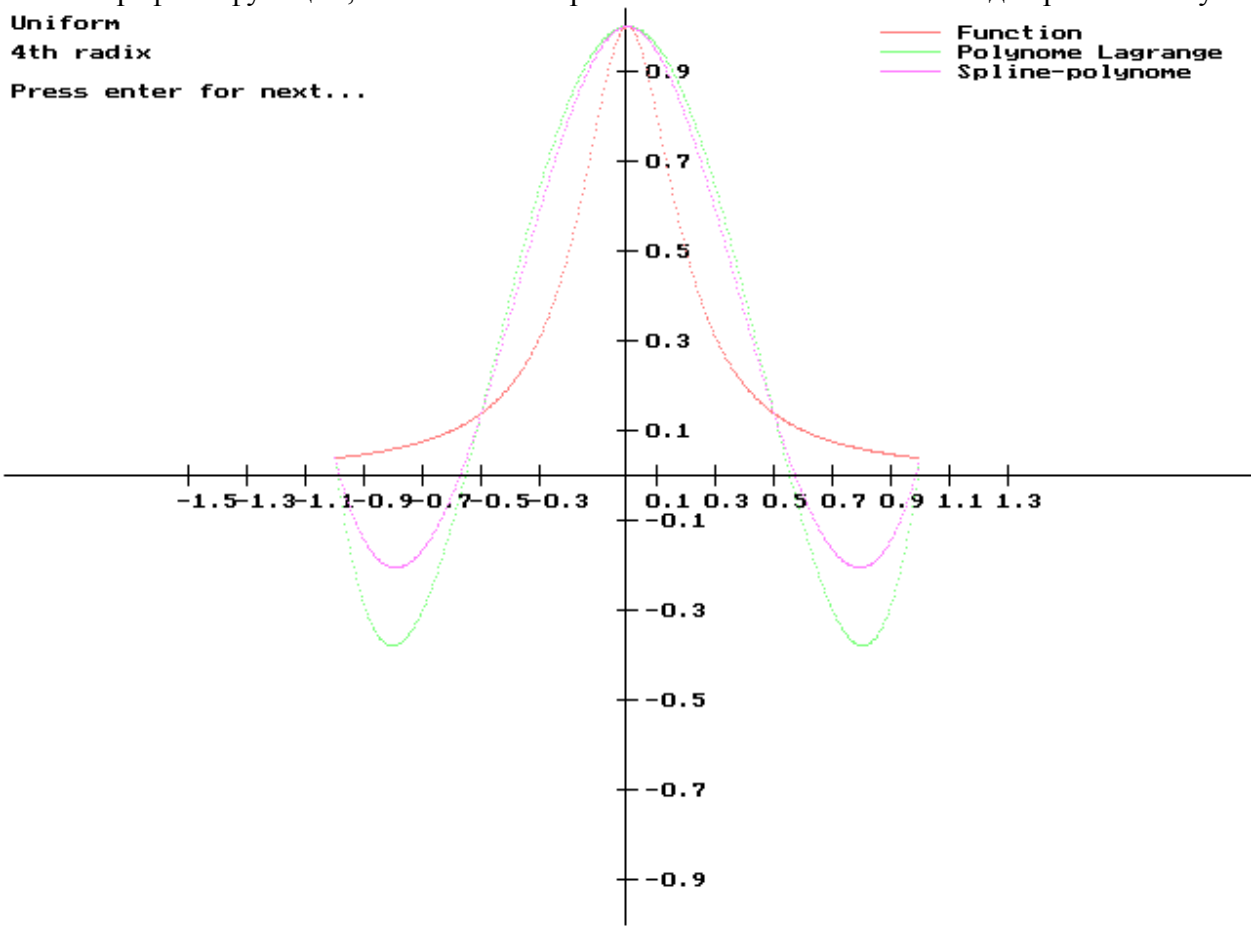
Press enter for next...

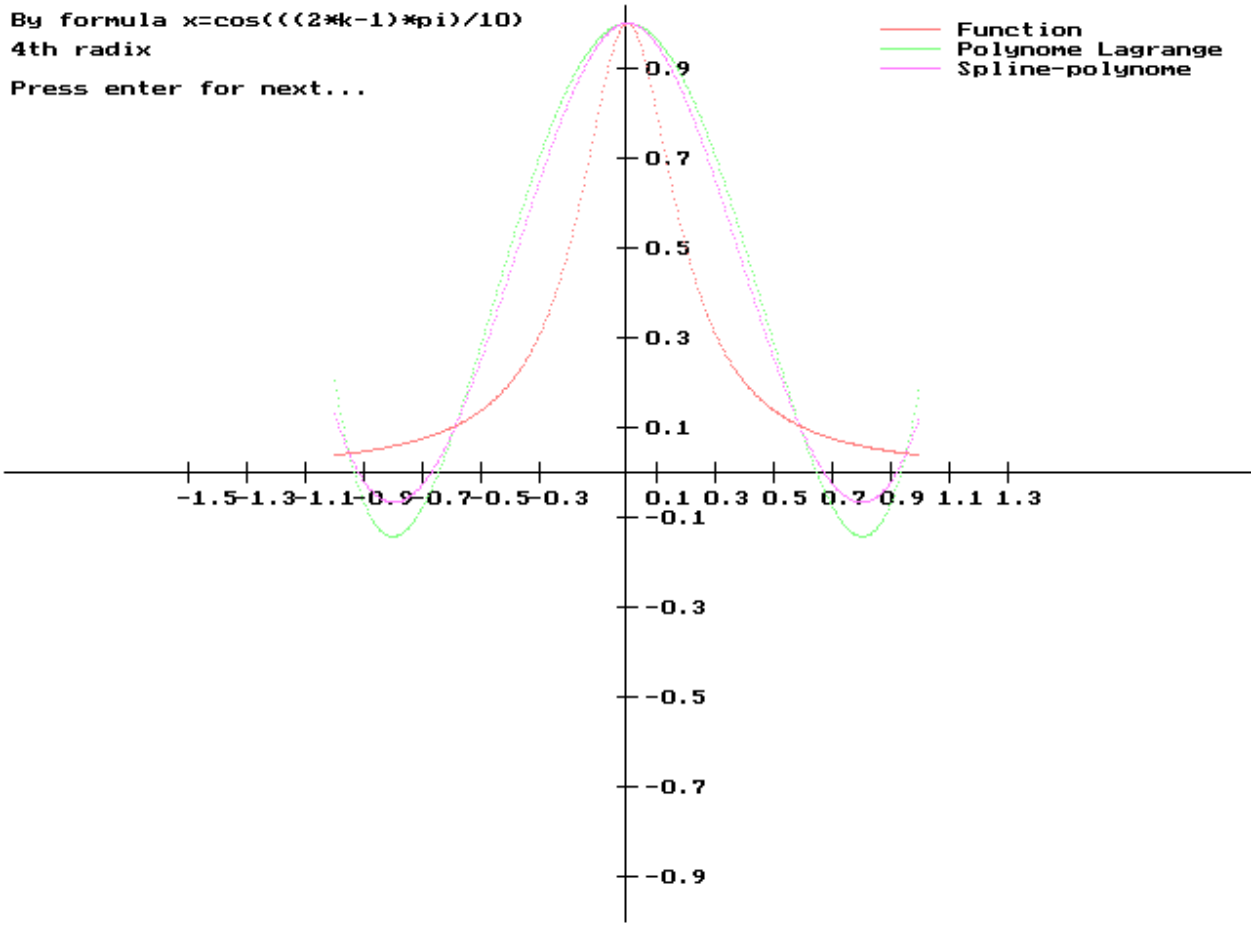
By formula x=cos(((2*K-1)*pi)/18) 8th radix

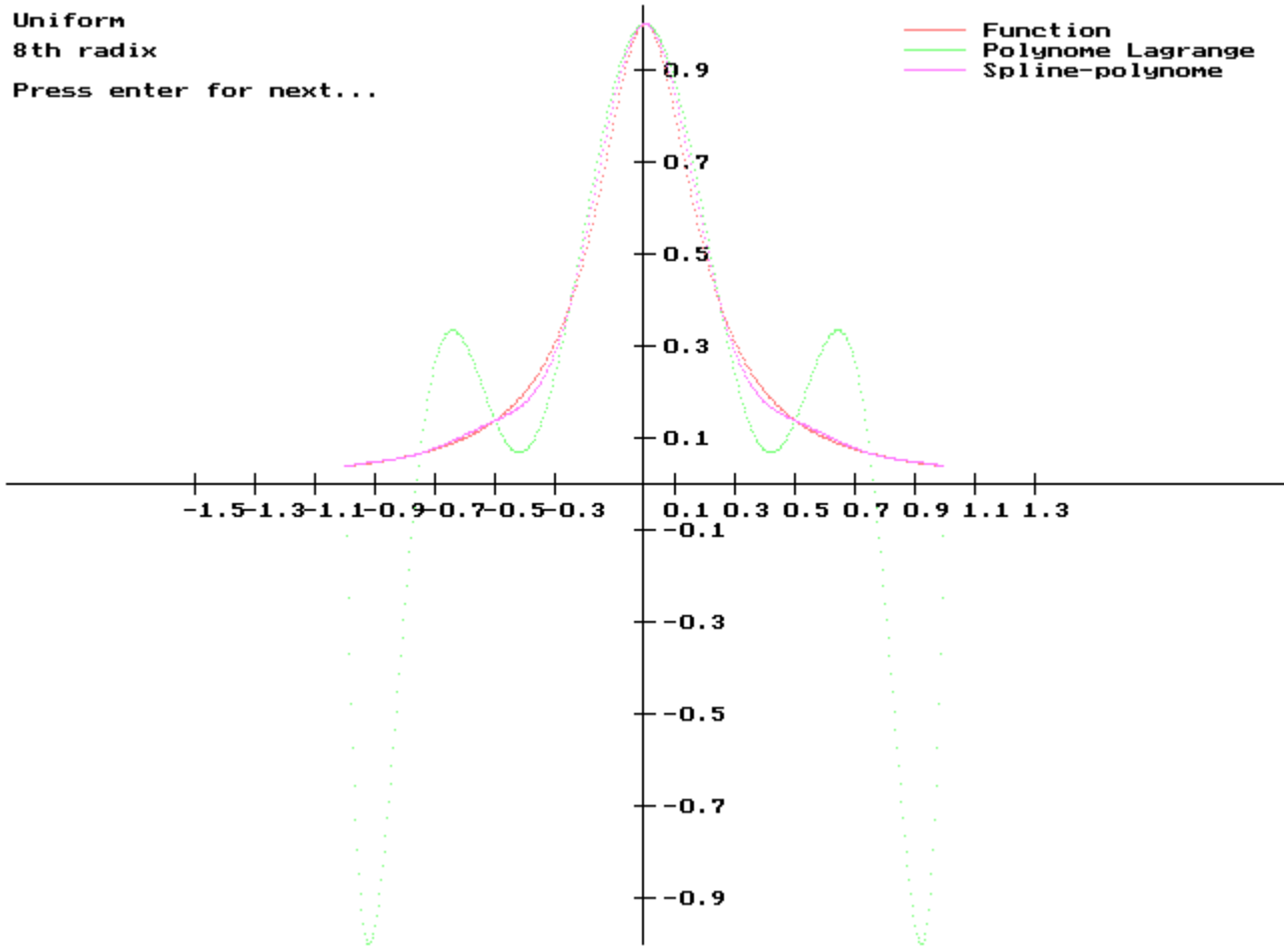
x	Lagrange	Spline
-1.000000	0.104643	0.039149
-0.900000	0.002772	0.045907
-0.800000	0.130207	0.063348
-0.700000	0.133714	0.082766
-0.600000	0.051258	0.088275
-0.500000	0.019260	0.096757
-0.400000	0.126638	0.164826
-0.300000	0.369452	0.349220
-0.200000	0.667083	0.628845
-0.100000	0.907960	0.886731
0.000000	1.000000	1.000000
0.100000	0.907960	0.886731
0.200000	0.667083	0.628845
0.300000	0.369452	0.349220
0.400000	0.126638	0.164826
0.500000	0.019260	0.096757
0.600000	0.051258	0.088275
0.700000	0.133714	0.082766
0.800000	0.130207	0.063348
0.900000	0.002772	0.045907
1.000000	0.104643	0.039149

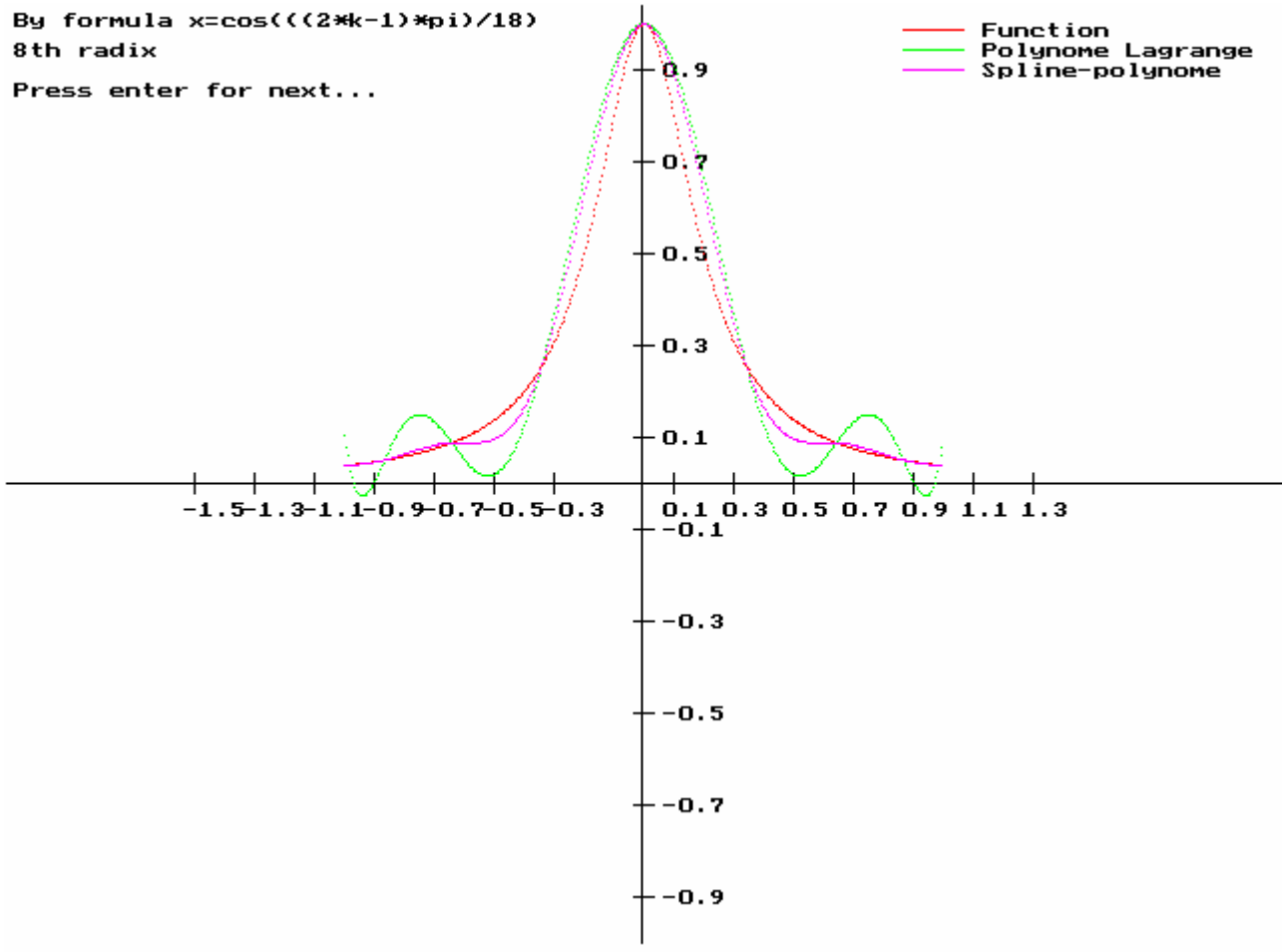
Press enter for next...

- Графики функции, полиномов Лагранжа и сплайнов 4-8 степени для различных узлов:






Uniform
8th radix
Press enter for next...
Function
Polynome Lagrange
Spline-polynome
0.9
0.7
0.5
0.3
0.1
-1.5 -1.3 -1.1 -0.9 -0.7 -0.5 -0.3
0.1 0.3 0.5 0.7 0.9 1.1 1.3
-0.1
-0.3
-0.5
-0.7
-0.9


By formula x=cos(((2*k-1)*pi)/18)
8th radix
Press enter for next...
Function
Polynome Lagrange
Spline-polynome
0.9
0.7
0.5
0.3
0.1
-1.5 -1.3 -1.1 -0.9 -0.7 -0.5 -0.3
0.1 0.3 0.5 0.7 0.9 1.1 1.3
-0.1
-0.3
-0.5
-0.7
-0.9

4. Вывод.

Из построенных графиков видно, что интерполяционные кривые в зависимости от количества узлов и их распределения приближены к графику функции с разной точностью. Чем ближе графики полиномов к графику функции, тем больше точность интерполяции. Из графиков видно, что наибольшая точность достигнута при построении полиномов на 8-ми узлах.