МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования   
**«Национальный исследовательский   
Нижегородский государственный университет им. Н.И. Лобачевского»**

**(ННГУ)**

**Институт информационных технологий, математики и механики**

**Кафедра математического обеспечения и суперкомпьютерных технологий**

Направление подготовки «Прикладная математика и информатика»

**ОТЧЕТ**

по лабораторной работе

**Табулирование функций**

**Выполнил:** студент группы 0813-1

В.И. Репин

**Проверил:**

Доцент, к.ф.-м.н.

Баркалов А.В.

Нижний Новгород  
2016

Содержание

Введение . . . . . . . . . 3

1. Постановка учебно-практической задачи . . . 3

2. Руководство пользователя . . . . . 4

3. Руководство программиста . . . . . 4

3.1. Описание алгоритмов . . . . . 4

3.2. Описание структур данных . . . . 5

3.3. Описание структуры программного комплекса . 6

Заключение . . . . . . . . . 8

Список литературы . . . . . . . 8

Приложение . . . . . . . . 8

**Введение**

Табулирование функции — это вычисление значений функции при изменении аргумента от некоторого начального значения до некоторого конечного значения с определённым шагом. Необходимость в табулировании возникает при решении достаточно широкого круга задач. Например, при численном решении нелинейных уравнений f(x) = 0, путём табулирования можно локализовать корни уравнения. С помощью табулирования можно найти минимум или максимум функции. Необходимость в табулировании возникает также при построении графиков функции на экране компьютера.

В отчёте приводится постановка задачи, описание использующихся алгоритмов, описание программы и правила её использования, а также прилагается код программы, решающей поставленную задачу.

**1. Постановка учебно-практической задачи**

*Формулировка задачи:*

Требуется разработать диалоговую многофайловую программу табулирования функций (exp, sin, cos, arctg), алгоритм вычисления которых задаётся отрезком ряда Тейлора.

Должна обеспечивать задание расчетных параметров:

* Функция, из числа реализованных;
* Границы изменения аргумента;
* Шаг изменения аргумента;
* Количество суммируемых членов ряда;
* Точность вычислений.

*Контрольный пример (рис. 1 и 2):*

******

Рис . 1 Исходные данные

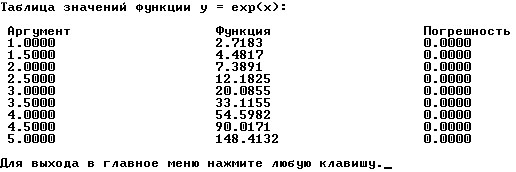
**

Рис . 2 Результат работы

**2. Руководство пользователя**

Для выполнения лабораторной работы по табулированию функций необходимо запустить на выполнение файл **Tabulation.exe**. На экране появится главное меню *(рис. 3).*

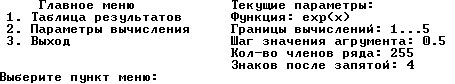


Рис . 3 Главное меню программы

Для получения таблицы значений функции с текущими параметрами необходимо выбрать **1** пункт меню. Результат представлен на *рис. 2.*

Для выхода из программы необходимо выбрать пункт **3** .

**2** пункт главного меню открывает меню задания параметров вычисления *(рис. 4):*

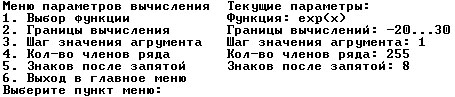
****

Рис . 4 Меню параметров вычисления

Пункты **1-5**  данного меню позволяют изменить значения параметров в определённых диапазонах значений. После изменения данные слева от меню обновляются.

Пункт **6** возвращает пользователя в главное меню**.**

**3. Руководство программиста**

* 1. **Описание используемых алгоритмов**

1. Приближённое вычисление значения функции с помощью ряда Тейлора

Суммируется ограниченное количество членов ряда, сходящегося к значению вычисляемой функции в данной точке. Каждый следующий член вычисляется рекуррентно. При этом функция побочным эффектом возвращает погрешность вычисления.

**3.2 Описание структур данных и типов**

Для реализации программы были созданы следующие структуры данных:

const

maxpr = 8; {Максимально возможная точность}

maxsteps = 30000; {Максимально возможное кол-во строк в таблице значений}

n\_main\_menu = 3; {Количество пунктов главного меню}

main\_menu\_list: array[1..n\_main\_menu] of string = (

'Таблица результатов',

'Параметры вычисления',

'Выход');

n\_set\_menu = 6; {Количество пунктов меню настроек}

set\_menu\_list: array[1..n\_set\_menu] of string = (

'Выбор функции',

'Границы вычисления',

'Шаг значения агрумента',

'Кол-во членов ряда',

'Знаков после запятой',

'Выход в главное меню');

n\_funct = 4; {Количество функций}

funct\_list: array [1.. n\_funct] of string = (

'exp(x)',

'sin(x)',

'cos(x)',

'arctg(x)');

def\_set\_list: array[1..n\_funct, 1..3] of real = {Максимальный диапазон, шаг по умолчанию для четырёх функций}

((-20, 30, 1),

(-25, 25, 0.5),

(-25, 25, 0.5),

(-1, 1, 0.01));

type

dif = function(x: real; n: byte; var c\_err: real): real; {Процедурный тип для выбора функции}

**3.3 Описание структуры программного комплекса.**

1. Основной файл программного кода, tabulation.pas.

{Вычисление значений и вывод таблицы}

procedure draw\_table(

f: dif; {Выбранная функция}

fn: byte; {Номер ф-ии в списке}

l, { Начальное значение аргумента}

r, {Конечное значение аргумента}

s:real; {Шаг аргумента}

n: byte; {Слагаемых Ряда Тейлора}

d: byte); {Точность}

{Вывод информации справа меню}

procedure draw\_info(f: byte; l, r, s: real; n: byte; d: byte); {Параметры аналогичны}

{Главное меню}

function main\_menu(f: byte; l, r, s: real; n: byte; d: bytе) {Параметры аналогичны}

: char; {Выбранный пункт меню}

{Меню параметров вычислений}

function set\_menu(f: byte; l, r, s: real; n: byte; d: bytе) {Параметры аналогичны}

: char; {Выбранный пункт меню}

{Меню выбора вычисляемой функции}

function funct\_choice\_menu

: char; {Выбранный пункт меню}

Далее следует основная программа, целиком состоящая из диалоговой системы и проверок валидности введённых данных.

1. Дополнительный файл программного кода (unit), tabul\_add.pas.

Модуль имеет следующий интерфейс:

var

err: boolean; { Для проверки

input\_val: string; ошибки во вводе}

pmenu: char; {Выбор пукта меню}

fchoice: dif; {Функция}

fnum: byte; {Номер функции в списке}

lside: real; {Левая граница для вычислений}

rside: real; {Правая граница для вычислений}

lrange: real; {Минимально возм. арг.}

rrange: real; {Максимально возм. арг.}

step: real; {Шаг арг.}

n\_row: byte; {Количество членов ряда}

pr: byte; {Точность}

tval: integer; {Вспомогательные

hv, hv1: real; переменные}

function a\_exp(

x: real; {Аргумент}

n: byte; {Кол-во членов ряда}

var c\_err: real) {Погрешность}

: real;

{Значение функции}

function a\_sin(x: real; n: byte; var c\_err: real): real; {Параметры аналогичны}

function a\_cos(x: real; n: byte; var c\_err: real): real; {Параметры аналогичны}

function a\_arctg(x: real; n: byte; var c\_err: real): real; {Параметры аналогичны}

{Формирует часть строки из таблицы значений}

function strform(

x: real; {Значение}

d: byte) {Точность}

: string; {Строка для вывода}

{Округление с заданной точностью}

function prec\_round(

var x: real; {Значение}

prec: byte) {Точность}

: string; {Строка на вывод (исп. в выводе текущих параметров)}

* Функции и процедуры, недоступные из основной программы:

{Исправляет аргументы для sin x, cos x не входящие в [0 , 2\*pi]}

procedure fix\_angle(

var x: real); {Значение}

Заключение

В лабораторной работе были реализованы алгоритмы приближённого вычисления четырёх функций, программный комплекс позволяет в режиме диалога изменять параметры вычислений и получить соответствующую таблицу значений требуемой функции. Причём при написании программы пришлось учесть, что способ приближения через ряд Тейлора не даёт верного результата для больших или отрицательных значений sin и cos (исправимо) и для arctg при аргументах |x|>1 (нельзя исправить). При этом приближенные снизу по модулю к единице аргументы дают большую погрешность даже при значительном количестве членов ряда. То есть этот ряд сходится к значению гораздо медленнее других.

**Список литературы**

1. В. Фаронов Turbo Pascal -СПб.: БХВ-Петербург, 2006.
2. Ильин В. А., Садовничий В. А., Сендов Б. Х. Математический анализ, ч. 1, изд. 3, ред. А. Н. Тихонов. М.: Проспект, 2004

**Приложение**

**Tabulation.pas**

program tabulation;

uses

crt,tabul\_add;

//-----------------Вычисление значений и вывод таблицы-------------------//

procedure draw\_table(

f: dif; //Выбранная функция

fn: byte; //Номер ф-ии в списке

l, // Начальное значение аргумента

r, // Конечное значение аргумента

s:real; // Шаг аргумента

n: byte; // Слагаемых Ряда Тейлора

d: byte); //Точность

var

calc\_err, cur: real;

i: word;

begin

clrscr;

writeln('Таблица значений функции y = ', funct\_list[fn], ':');

writeln;

writeln(' Аргумент Функция Погрешность');

cur := l;

for i := 1 to round((r - l) / step) do

begin

writeln(' ', strform(cur, d), strform(f(cur, n, calc\_err), d), strform(calc\_err, d));

cur += step;

end;

writeln(' ', strform(r, d), strform(f(r, n, calc\_err), d), strform(calc\_err, d));

writeln;

write('Для выхода в главное меню нажмите любую клавишу.');

readln;

end;

//-----------------------------------------------------------------//

//---------------------Вывод информации справа меню----------------//

procedure draw\_info(

f: byte; //Номер ф-ии в списке

l, //Нач. зн. арг.

r, // Кон. зн. арг.

s: real; //Шаг арг.

n: byte; // Слаг. в Ряду Тейлора

d: byte); //Точность

var

sh: byte;

begin

sh := 30;

gotoxy(sh, 1);

write('Текущие параметры: ');

gotoxy(sh, 2);

write('Функция: ', funct\_list[f]);

gotoxy(sh, 3);

write('Границы вычислений: ', prec\_round(l, pr), '...', prec\_round(r, pr));

gotoxy(sh, 4);

write('Шаг значения агрумента: ', prec\_round(s, pr));

gotoxy(sh, 5);

write('Кол-во членов ряда: ', n);

gotoxy(sh, 6);

write('Знаков после запятой: ', d);

writeln;

end;

//--------------------------------------------------------------//

//---------------------Главное меню-----------------------------//

function main\_menu(

f: byte; //Номер ф-ии в списке

l, //Нач. зн. арг.

r, // Кон. зн. арг.

s: real; //Шаг арг.

n: byte; // Слаг. в Ряду Тейлора

d: byte) //Точность

: char; //Возвращает выбранный пункт меню.

var

i: byte;

c: char;

begin

clrscr;

writeln(' Главное меню');

for i := 1 to n\_main\_menu do

writeln(' ', i, '. ', main\_menu\_list[i]);

draw\_info(f, l, r, s, n, d);

write( 'Выберите пункт меню: ');

readln(c);

main\_menu := c;

end;

//--------------------------------------------------------------//

//-------------------Меню параметров----------------------------//

function set\_menu(

f: byte; //Номер ф-ии в списке

l, //Нач. зн. арг.

r, // Кон. зн. арг.

s: real; //Шаг арг.

n: byte; // Слаг. в Ряду Тейлора

d: byte) //Точность

: char; //Возвращает выбранный пункт меню.

var

i: byte;

c: char;

begin

clrscr;

writeln(' Меню параметров вычисления');

for i := 1 to n\_set\_menu do

writeln(' ', i, '. ', set\_menu\_list[i]);

draw\_info(f, l, r, s, n, d);

writeln;

write( ' Выберите пункт меню: ');

readln(c);

set\_menu := c;

end;

//--------------Меню выбора функции---------------------//

function funct\_choice\_menu:

char; //Возвращает выбранный пункт меню

var

i: byte;

c: char;

begin

clrscr;

for i := 1 to n\_funct do

writeln(i, '. ', funct\_list[i]);

write('Выберите функцию: ');

readln(c);

funct\_choice\_menu := c;

end;

//-----------------------------------------------------//

//----------------Основная программа-------------------//

begin

//-----------Установка параметров по умолчанию-------//

fchoice := a\_exp;

fnum := 1;

lrange := def\_set\_list[fnum, 1];

rrange := def\_set\_list[fnum, 2];

lside := lrange;

rside := rrange;

step := def\_set\_list[fnum, 3];

n\_row := 255;

pr := maxpr;

//--------------------------------------------------//

//---------------------Диалоговый раздел------------//

repeat

pmenu := main\_menu(fnum, lside, rside, step, n\_row, pr);

case pmenu of

'1': draw\_table(fchoice, fnum, lside, rside, step, n\_row, pr);

'2':

repeat

pmenu := set\_menu(fnum, lside, rside, step, n\_row, pr);

case pmenu of

'1':

begin

repeat

pmenu := funct\_choice\_menu;

case pmenu of

'1': begin fchoice := a\_exp; fnum := 1; end;

'2': begin fchoice := a\_sin; fnum := 2; end;

'3': begin fchoice := a\_cos; fnum := 3; end;

'4': begin fchoice := a\_arctg; fnum := 4; end;

else

begin

write('Неверный пункт меню. Повторите ввод.');

readln;

end;

end;

until not (err);

lrange := def\_set\_list[fnum, 1];

rrange := def\_set\_list[fnum, 2];

lside := lrange;

rside := rrange;

step := def\_set\_list[fnum, 3];

end;

'2':

begin

repeat

clrscr;

writeln('Введите левую границу: [', lrange, ' .. ', rrange, '), с точностью до ', pr, '^-10: ');

readln(input\_val);

err := not (trystrtofloat(input\_val, lside));

prec\_round(lside, pr);

if (lside >= rrange) or (lside < lrange) then

err := true;

if err then

begin

write('Неверное значение, повторите ввод');

readln;

end;

until not (err);

repeat

clrscr;

writeln('Введите правую границу (', prec\_round(lside, pr), ' .. ', rrange, '], с точностью до ', pr, '^-10: ');

readln(input\_val);

err := not (trystrtofloat(input\_val, rside));

prec\_round(rside, pr);

if (rside <= lside) or (rside > rrange) then

err := true;

if err then

begin

write('Неверное значение, повторите ввод');

readln;

end;

until not (err);

hv := rside - lside;

prec\_round(hv, pr);

if step > hv then step := hv;

end;

'3':

begin

hv := rside - lside;

hv1 := hv / maxsteps;

repeat

clrscr;

writeln('Введите шаг значения аргумента (', prec\_round(hv1, pr), ' .. ', prec\_round(hv, pr), '], с точностью до ', pr, '^-10: ');

readln(input\_val);

err := not (trystrtofloat(input\_val, step));

prec\_round(step, pr);

if (step <= hv1) or (step > hv) then

err := true;

if err then

begin

write('Неверное значение, повторите ввод');

readln;

end;

until not (err);

end;

'4':

begin

repeat

clrscr;

write('Введите количество членов отрезка ряда 1 .. 255: ');

readln(input\_val);

err := not (trystrtoint(input\_val, tval));

if tval <= 0 then

err := true;

if err then

begin

write('Неверное значение, повторите ввод');

readln;

end;

until not (err);

n\_row := tval;

end;

'5':

begin

repeat

clrscr;

write('Введите количество знаков после запятой 3 .. ', maxpr, ': ');

readln(input\_val);

err := not (trystrtoint(input\_val, tval));

if (tval > maxpr) or (tval < 3) then

err := true;

if err then

begin

write('Неверное значение, повторите ввод');

readln;

end;

until not (err);

pr := tval;

end;

'6': ;

else

begin

write('Неверный пункт меню. Повторите ввод.');

readln;

end;

end;

until ord(pmenu) - ord('0') = n\_set\_menu;

'3': ;

else

begin

write('Неверный пункт меню. Повторите ввод.');

readln;

end;

end;

until ord(pmenu) - ord('0') = n\_main\_menu;

end.

**Tabul\_add.pas**

unit tabul\_add;

interface

const

maxpr = 8; //Максимально возможная точность

maxsteps = 30000; //Максимально возможное кол-во строк в таблице значений

n\_main\_menu = 3; //Количество пунктов главного меню

main\_menu\_list: array[1..n\_main\_menu] of string = (

'Таблица результатов',

'Параметры вычисления',

'Выход');

n\_set\_menu = 6; //Количество пунктов меню настроек

set\_menu\_list: array[1..n\_set\_menu] of string = (

'Выбор функции',

'Границы вычисления',

'Шаг значения агрумента',

'Кол-во членов ряда',

'Знаков после запятой',

'Выход в главное меню');

n\_funct = 4; //Количество функций

funct\_list: array [1.. n\_funct] of string = (

'exp(x)',

'sin(x)',

'cos(x)',

'arctg(x)');

def\_set\_list: array[1..n\_funct, 1..3] of real = //(Максимальный диапазон, шаг по умолчанию) для четырёх функций

((-20, 30, 1),

(-25, 25, 0.5),

(-25, 25, 0.5),

(-1, 1, 0.01));

type

dif = function(x: real; n: byte; var c\_err: real): real;//Процедурный тип для выбора функции

var

err: boolean; // Для проверки

input\_val: string; // ошибки во вводе

pmenu: char; //Выбор пукта меню

fchoice: dif; //Функция

fnum: byte; //Номер функции в всписке

lside: real; //Левая граница для вычислений

rside: real; //Правая граница для вычислений

lrange: real; //Минимально возм. арг.

rrange: real;// Максимально возм. арг.

step: real; //Шаг арг.

n\_row: byte; //Количество членов ряда

pr: byte; //Точность

tval: integer; // Вспомогательные

hv, hv1: real;// переменные

function a\_exp(x: real; n: byte; var c\_err: real): real;

function a\_sin(x: real; n: byte; var c\_err: real): real;

function a\_cos(x: real; n: byte; var c\_err: real): real;

function a\_arctg(x: real; n: byte; var c\_err: real): real;

function strform(x: real; d: byte): string;

function prec\_round(var x: real; prec: byte): string;

implementation

//----------Формирует часть строки из таблицы значений (некорректная работа :a:b)--//

function strform(

x: real; //Значение

d: byte) //Точность

: string; //Строка для вывода

var

res: string;

i: byte;

begin

str(x:26:d, res);

res := trim(res);

for i := 1 to 26 - length(res) do

res += ' ';

strform := res;

end;

//------------------------------------------------------------------------------//

//-----------------Округление с заданной точностью (некорректные результаты вычислений и сравнений)----//

function prec\_round(

var x: real; //Значение

prec: byte) //Точность

: string; //Строка на вывод (опционально)

var

temp: string;

i: byte;

begin

str(x:3 + prec:prec, temp);

temp := trim(temp);

i := length(temp);

while temp[i] = '0' do

begin

delete(temp, i, 1);

dec(i);

end;

if temp[i] = '.' then delete(temp, i, 1);

x := strtofloat(temp);

prec\_round := temp;

end;

//-----------------------------------------------------------------------//

//-------Исправляет аргументы для sin x, cosx не входящие в [0 , 2\*pi]---//

procedure fix\_angle(

var x: real);//Значение

begin

while x < 0 do

x += 2 \* pi;

while x > 2\*pi do

x -= 2 \* pi;

end;

//-------------------------------------------------//

//--------------Вычисление exp(x)------------------//

function a\_exp(x: real; n: byte; var c\_err: real): real;

var

i: byte;

sum, cur: real;

begin

cur := 1;

for i := 1 to n do

begin

sum += cur;

cur \*= x / i;

end;

c\_err := abs(exp(x) - sum);

a\_exp := sum;

end;

//-------------------------------------------------//

//--------------Вычисление sin(x)------------------//

function a\_sin(x: real; n: byte; var c\_err: real): real;

var

i: byte;

sum, cur: real;

begin

fix\_angle(x);

cur := x;

sum := cur;

for i := 2 to n do

begin

cur \*= -x \* x / ((2 \* i - 2) \* (2 \* i - 1));

sum += cur;

end;

c\_err := abs(sin(x) - sum);

a\_sin := sum;

end;

//-------------------------------------------------//

//--------------Вычисление cos(x)------------------//

function a\_cos(x: real; n: byte; var c\_err: real): real;

var

i: byte;

sum, cur: real;

begin

fix\_angle(x);

cur := 1;

for i := 1 to n do

begin

sum += cur;

cur \*= -x \* x /( (2 \* i)\*(2\*i-1));

end;

c\_err := abs(cos(x) - sum);

a\_cos := sum;

end;

//---------------------------------------------------//

//--------------Вычисление arctg(x)------------------//

function a\_arctg(x: real; n: byte; var c\_err: real): real;

var

i: byte;

sum, cur: real;

begin

cur := x;

sum := cur;

for i := 2 to n do

begin

cur \*= -x \* x / (2 \* i - 1);

sum += cur;

end;

c\_err := abs(arctan(x) - sum);

a\_arctg := sum;

end;

//--------------------------------------------------//

begin

end.