Лабораторная работа №4. Детерминированные вычислительные процессы с управлением по аргументу. Численное интегрирование с использованием функции пользователя.

Цель лабораторной работы: изучение принципов использования ДЦВП с управлением по аргументу при помощи функций средствами Lazarus и Free Pascal.

Оборудование: ПК, Lazarus, Word, Draw.io.

Задание 1.

1. Реализовать вычисление определенного интеграла из индивидуального задания (взять интеграл из предыдущей лабораторной) методом трапеций с использованием пользовательской функции.
2. Математическая модель:



1. Блок-схема:



1. Список идентификаторов:

|  |  |  |
| --- | --- | --- |
| Имя | Смысл | Тип |
| a | Левая граница интегрируемого интервала | real |
| b | Правая граница интегрируемого интервала | real |
| N | Количество разбиений интервала | word |
| k | Переменная-счётчик, служит аргументом для вычисления функции | real |
| h | Величина шага | real |
| x | Аргумент функции func | real |
| func | Значение, возвращаемое функцией func при расчёте значения подынтегральной функции в данной точке интегрируемого интервала | real |
| res | Результат | real |

1. Код программы:

program trapezium;

var a, b, h, res, k: real;

N: word;

function func(x:real):real;

begin

func:=(sin(x+1.4))/(0.8+cos(2\*x\*x+0.5));

end;

begin

writeln('Enter N');

readln(N);

a:= 0.4;

b:= 1.0;

h:=(b-a)/N;

res:= (func(a)+func(b))/2;

k:= a+h;

while k <= b-h do begin

res:= res+func(k);

k:=k+h;

end;

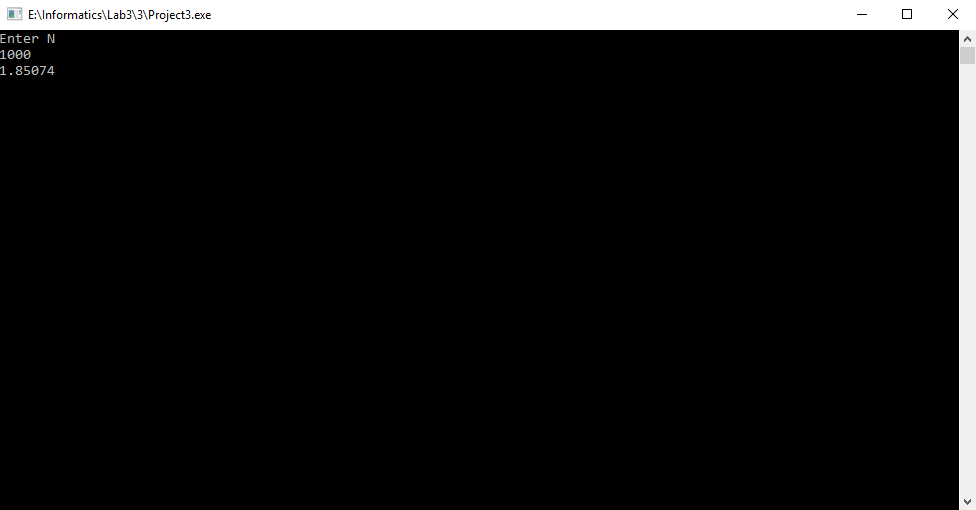
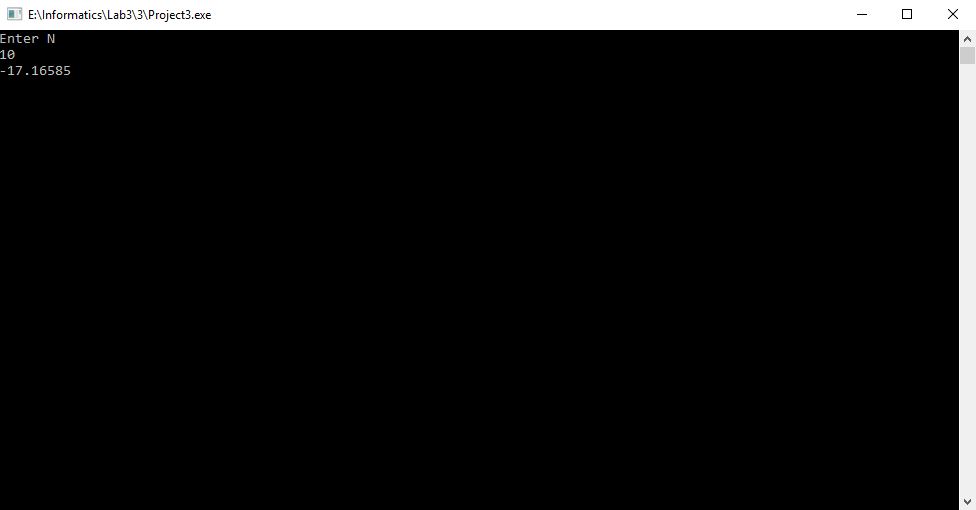
res:=h\*res;

writeln(res:5:5);

readln();

end.

1. Результаты вычислений



1. Анализ результатов вычисления:

Программа поддерживает до 65535 разбиений интегрируемого интервала. Цикл вычислений проходит от a + h до b – h с использованием функции sum, работающей на входе с переменной-счётчиком k, также с её помощью вычисляются значения подынтегральной функции на границах интегрируемого интервала a и b. Точность нахождения определенного интеграла определяется количеством разбиений N.

Задание 2.

1. Реализовать вычисление определенного интеграла из индивидуального задания (взять интеграл из предыдущей лабораторной) методом парабол с использованием пользовательской функции.



1. Математическая модель:

1. Блок-схема:





1. Список идентификаторов:

|  |  |  |
| --- | --- | --- |
| Имя | Смысл | Тип |
| a | Левая граница интегрируемого интервала | real |
| b | Правая граница интегрируемого интервала | real |
| N | Количество разбиений интервала | word |
| h | Величина шага | real |
| s | Сумма значений первого цикла, затем - второго | real |
| res | Результат | real |
| x | Аргумент функции func | real |
| k | Переменная-счётчик для подсчёта сумм значений элементов с нечётными и чётными индексами в первом и втором цикле соответственно | real |
| func | Значение, возвращаемое функцией func при расчёте значения подынтегральной функции в данной точке интегрируемого интервала | real |

1. Код программы:

program parabola;

var a, b, h, s, res, k: real;

N: word;

function func(x:real):real;

begin

sum:=(sin(x+1.4))/(0.8+cos(2\*x\*x+0.5));

end;

begin

a:= 0.4;

b:= 1;

writeln('Enter N');

readln(N);

h:=(b-a)/N;

res:=func(a)+func(b);

k:= a+h;

s:=0;

while k <= b-h do begin

s:= s + func(k);

k:= k + 2\*h;

end;

res:=res+s\*4;

s:=0;

k:= a+2\*h;

while k <= b-2\*h do begin

s:= s + func(k);

k:= k + 2\*h;

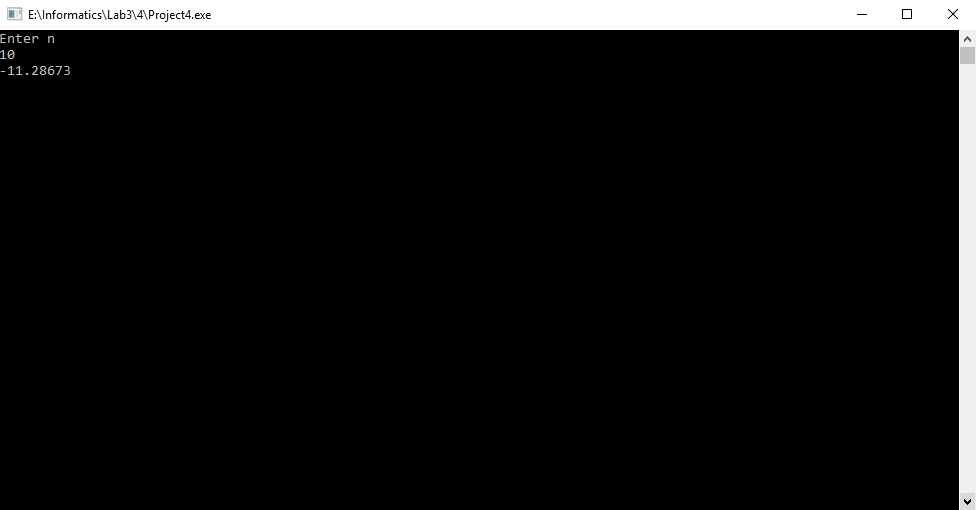
end;

res:= h\*(res+s\*2)/3;

writeln(res:5:5);

readln();

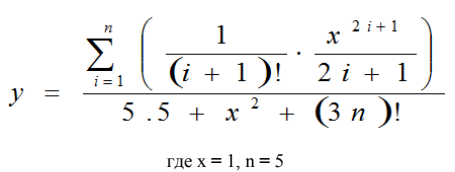
end.

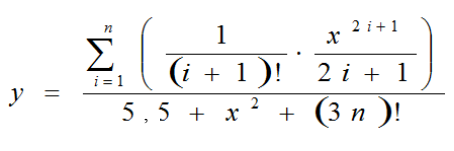
1. Результаты вычислений: 
2. Анализ результатов вычисления:

Программа поддерживает до 65535 разбиений интегрируемого интервала. После вычисления величины шага происходит подсчёт суммы y(a) и y(b) с использованием функции func. В первом цикле подсчитывается сумма y с нечётными индексами от a + h до b – h. Полученная сумма записывается в переменную s, значение которой затем присваивается переменной res и обнуляется для подсчёта суммы значений второго цикла. Во втором цикле суммируются значения y с чётными индексами от a + 2\*h до b – 2\*h. Оба цикла используют функцию func, использующую в качестве аргумента переменную-счётчик k.

Задание 3.

1. Вычислить:



1. Математическая модель: 
2. Блок-схема:



1. Список идентификаторов:

|  |  |  |
| --- | --- | --- |
| Имя | Смысл | Тип |
| x | Аргумент исходной функции, задаётся пользователем | real |
| n | Аргумент исходной функции, задаётся пользователем | byte |
| res | Переменная, хранящая итоговый результат вычислений | real |
| fact | Идентификатор функции, расчитывающей факториал данного числа | longint |
| upsd | Идентификатор функции, расчитывающей значение суммы от i до n исходной функции | real |
| y | Параметр цикла for, используемого в теле функции fact | byte |
| i | Паарметр цикла for, используемого в теле функции upsd | byte |

1. Код программы:

program garbage;

var n:byte;

x,res:real;

function fact(n:byte):longint;

var y:byte;

begin

fact:=1;

for y:=2 to n do

fact:=fact\*y;

end;

function upsd(n:byte;x:real):real;

var i:byte;

p\_res:word;

begin

upsd:=0;

for i:=1 to n do begin

p\_res:=2\*i+1;

upsd:=upsd+exp(p\_res\*ln(x))/(fact(i+1)\*p\_res);

end;

end;

begin

writeln('Введите n, x');

readln(n,x);

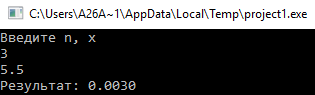
res:=upsd(n,x)/(5.5+exp(2\*ln(x))+fact(3\*n));

writeln(res:3:4);

readln();

end.

1. Результаты выполненной работы:



1. Анализ результатов вычисления:

В программе описаны две пользовательские функции: fact и upsd, предназначенные для вычисления факториала и выражения в числителе соответственно. Функция fact принимает на вход параметр n, и при помощи цикла for с параметром y от 2 до n находит факториал заданного числа. Функция upsd принимает на вход два параметра n и x. При помощи цикла for с параметром i от 1 до n ведётся расчёт суммы исходной функции, расположенной под знаком суммы.

**Вывод**:

Использование функций в программах с повторяющимися блоками инструкций позволяет упростить код программы, сократив его длину и сделать его более разборчивым. В ряде случаев также повышается эффективность программ.