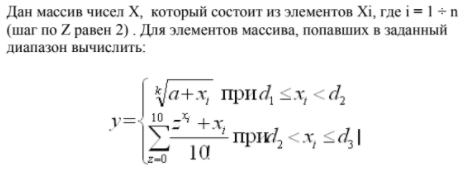
Лабораторная работа №11. Комбинированные вычислительные процессы.

Цель: рассмотрение способов практического применения комбинированных вычислительных процессов и особенностей их организации.

Оборудование: ПК, Lazarus, Word, Draw.io

Задание 1.

1. 
2. Математическая модель:
3. Блок-схема:



1. Список идентификаторов:

|  |  |  |
| --- | --- | --- |
| Имя | Смысл | Тип |
| b | Массив аргументов, заполняемый пользователем с клавиатуры | shortint |
| i | Параметр циклов for, также переменная, в которую записывается значение, вводимое пользователем при выборе изменяемой величины в случае несоответствия подкоренного выражения ОДЗ | byte |
| z | Исходная переменная | byte |
| n | Переменная, хранящая вводимое пользователем значение длины массива | byte |
| k | Вводимое пользователем значение, определяющая показатель степени 1/k | real |
| d1 | Граница диапазона | real |
| d2 | Граница диапазонов    и | real |
| d3 | Граница диапазона | real |
| a | Вводимое пользователем значение, параметр подкоренного выражения | real |
| y | Выходное значение функции | double |
| fact | Переменная, хранящая значение факториала, вычисленного на данной итерации | longint |

1. Код программы:

program one;

uses math;

var b:array of shortint;

i,z,n:byte;

k,d1,d2,d3,a:real;

y:double;

fact:longint;

begin

writeln('Enter d1,d2,d3');

readln(d1,d2,d3);

writeln('Enter a');

readln(a);

while a+d1<0 do begin

writeln('Введённые значения a и d1 не соответствуют ОДЗ. Измените a или d1.');

writeln('Введите 1 для изменения a или 2 для изменения границы интервала d1');

readln(i);

if i=1 then begin

writeln('Enter a');

readln(a);

end

else if i=2 then begin

writeln('Enter d1');

readln(d1);

end;

end;

writeln('Enter k');

readln(k);

writeln('Enter array length');

readln(n);

setlength(b,n+1);

fact:=1;

for i:=2 to 10 do

fact:=fact\*i;

for i:=1 to n do begin

writeln('Enter element');

readln(b[i]);

if (b[i]<d2) and (b[i]>=d1) then begin

y:=power(a+b[i],1/k);

writeln('b[',i,'] = ',b[i]);

writeln('y = ',y:0:3);

end

else if (b[i]>d2) and (b[i]<=d3) then

begin

z:=0;

y:=0;

while z<=10 do begin

y:=y+(power(z,b[i])+b[i]);

z:=z+2;

end;

y:=y/fact;

writeln('b[',i,'] = ',b[i]);

writeln('y = ',y:0:3);

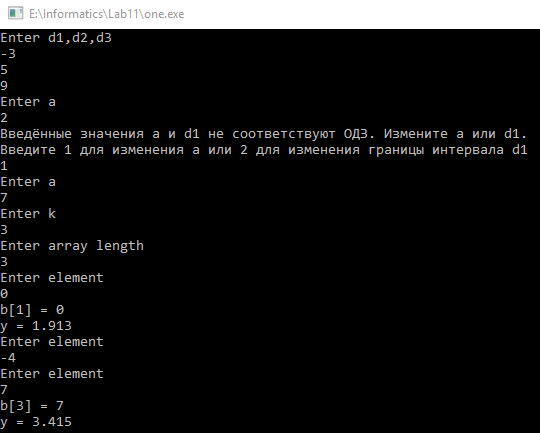
end;

end;

readln();

end.

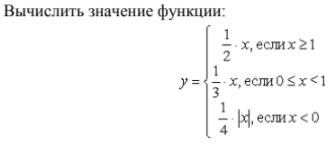
1. Результат выполненной работы:

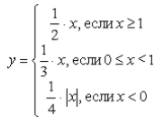


1. Анализ результатов вычисления:

После ввода границ диапазонов d1,d2,d3 и параметра подкоренного выражения a происходит проверка на соответствие ОДЗ подкоренного выражения, в котором роль слагаемого bᵢ выполняет d1. При получении отрицательного значения суммы a+d1 пользователю предалагется изменить по выбору a или d1. Проверка осущесвтлена в цикле while, и будет продолжаться пока не будет получено выражение >= 0. При соответствии данному условию происходит выход из цикла и дальнейший ввод значений k, n, где n – длина массива, вводимая пользователем. Т.к. по условию задания необходимо перебрать элементы с индексами от 1 до n, длина массива устанавливается равной n+1, в связи с тем, что индексация динамического массива начинается с 0. Далее в цикле for считается факториал 10, после чего начинается основной цикл, в котором вводятся и обрабатываются элементы массива. Выбор формулы для высчисления функции осуществляется при помощи конструкции if..else if..

Задание 2.

1. 
2. Математическая модель:



1. Блок-схема:

****

1. Список идентификаторов:

|  |  |  |
| --- | --- | --- |
| Имя | Смысл | Тип |
| x | Вводимое пользователем значение аргумента | real |
| y | Выходное значение функции | real |

1. Код программы:

program two;

var x,y:real;

begin

writeln('Enter x');

readln(x);

if x>=1 then

y:=x\*0.5

else if x<0 then

y:=abs(x)\*0.25

else

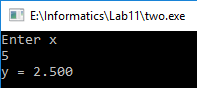
y:=x/3;

writeln('y = ',y:0:3);

readln();

end.

1. Резульат выполненной работы:



1. Анализ результатов вычисления:

При запуске программы пользователю предлагается ввести значение аргумента x, после чего происходит проверка на вхождение в заданные диапазоны при помощи конструкции if..else if..else. При определении ветви происходит вычисление значения функции по соответствующей формуле и последующий вывод полученного результата на экран.

Задание 3.

1. 
2. Математическая модель:
3. Блок-схема:



1. Список идентфиикаторов:

|  |  |  |
| --- | --- | --- |
| Имя | Смысл | Тип |
| a | Вводимое пользователем число | double |
| s | Сумма факториалов целых чисел | longint |
| fact | Факториал целого числа, расчитываемый на данной итерации | longint |
| i | Переменная для подсчёта факториала, увеличивающаяся на каждой итерации | byte |

1. Код программы:

program three;

var s,fact:longint;

i:byte;

a:real;

begin

writeln('Enter A >= 1');

readln(a);

while a<1 do begin

writeln('Inadmissible number, re-enter');

readln(a);

end;

s:=0;

fact:=1;

i:=0;

while 1=1 do begin

inc(i);

fact:=fact\*i;

s:=s+fact;

if s>a then begin

s:=s-fact;

break;

end

else if s=a then begin

writeln(i,' term = ',fact);

break;

end;

writeln(i,' term = ',fact);

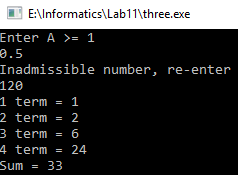
end;

writeln('Sum = ',s);

readln();

end.

1. Результат выполненной работы:



1. Анализ результатов вычисления:

При запуске программы пользователю предлагается ввести значение a, определяющее условие, до выполнения которого будут вестись расчёты. При вводе A < 1 выводится сообщение о недопустимом значении, т.к. сумма факториалов не может быть меньше 1. Далее запускается цикл while 1=1, в теле которого расчитывается факториал целого числа, прибавляющийся к сумме s. Происходит проверка на превышение заданного числа a, в случае истинности данного условия из суммы вычитается значение ранее прибавленного факториала, осуществляется выход из цикла командой break. Далее проверяется равенство суммы и значения a. В таком случае на экран выводится последнее слагаемое и происходит выход из цикла. Следующая команда – вывод на экран факториала-слагаемого. Таким образом подсчёты ведутся до тех пор, пока не будет установлено равенство переменных или превышение s>a.

Вывод:

Комбинированные вычислительные процессы, сочетающие в себе ранее рассмотренные линейные, итерационные циклические вычислительные процессы и т.д.находят применение в самых различных задачах. Организация данных процессов происходит при помощи имеющихся во Free Pascal конструкциях for, while, if, case..of и т.д.