**ОТЧЕТ**

по лабораторной работе №4

ПРОГРАММИРОВАНИЕ МЕТОДОВ ОБРАБОТКИ ОДНОМЕРНЫХ МАССИВОВ НА ЯЗЫКЕ PASCAL

Выполнил: студент группы ИВТ/б-12о

Горбенко Кирилл

Проверил: доцент кафедры ИУТС

Осадченко А.Е.

1. Цель работы

Исследование циклических алгоритмов и программ, осуществляющих типичные операции над одномерными массивами. Приобретение навыков программирования и вывода массивов.

1. Задание на работу

Создать программу, реализующую обработку массива вещественных чисел с использованием двух подпрограмм: процедуры и функции. Исходные данные для варианта №9:

Таблица 1

|  |  |  |
| --- | --- | --- |
| Задания и значения исходных данных | | |
| Набор значений элементов массива и дополнительные исходные данные | Функция должна вычислять | Процедура должна выводить на экран элементы, удовлетворяющие условию |
| X = 1; 15,2; -15; 109; -0,8; 0,17; -9\*10-4;1.  B = 1. | Сумму отрицательных элементов |  |

Для использования заданного массива в виде параметра процедуры и функции, создадим пользовательский тип данных – массив значений типа real с индексами в диапазоне от 1 до 8 (количество элементов массива):

**Type** massive = **array**[1..n] **of** real;

1. Схемы программ
   1. Схема основной программы

Начало

Ввод b; n = 8

i от 1 до n

n – количество элементов массива

Ввод mass[i]

sum\_of\_negative

(n, mass)

elements\_satisfying

(b, n, mass)

Конец

i = i + 1

Рисунок 1 – Блок-схема основной программы

* 1. Схема подпрограммы-функции, вычисляющей сумму отрицательных элементов массива

Начало

Чтение size и массива massive\_given

i от 1 до size

Нет

Да

i = i + 1

Конец

Рисунок 2 – Блок-схема подпрограммы-функции

* 1. Схема подпрограммы-процедуры, выводящей на экран все неотрицательные элементы массива

Начало

i = i + 1

Вывод

i от 1 до size

Чтение a, size и массива massive\_given

Да

Нет

Конец

Рисунок 3 – Блок-схема подпрограммы-процедуры

1. Текст программы

**program** lr4;

**const**

n = 8;

**type**

massive = **array**[1..n] **of** real;

**var**

b: integer;

i: byte;

mass: massive;

sum: real;

**procedure** elements\_satisfying(a: integer; size: byte; **const** massive\_given: massive);

**var**

i: byte;

**begin**

writeln('Результат работы процедуры: ');

**for** i := 1 **to** size **do**

**if** sqrt(abs(massive\_given[i])) < a **then** writeln(massive\_given[i]);

**end**;

**function** sum\_of\_negative(size : byte; **const** massive\_given: massive): real;

**var**

i: byte;

sum: real;

**begin**

**for** i := 1 **to** size **do**

**if** massive\_given[i] < 0 **then** sum := sum + massive\_given[i];

sum\_of\_negative := sum;

**end**;

**begin**

writeln('Введите параметр для использования в процедуре'); readln(b);

writeln('Вводите по очереди элементы массива');

**for** i := 1 **to** n **do**

read(mass[i]);

sum := sum\_of\_negative(n, mass);

writeln('Результат работы функции: ', sum);

elements\_satisfying(b ,n , mass);

**end**.

1. Результаты работы программы

Программа дает следующий результат:

* результат работы функции: -15.8009;
* результат работы процедуры: -0.8; 0.17; -0.0009.

1. Вывод

Использование функции оправдано в связи с необходимостью вернуть в программу единственное значение, в данном случае – вещественное число. Процедуры используются из-за необходимости выборки из массива и вывода на экран нескольких чисел-элементов массива.

Чтобы осуществить ввод массива в функцию и процедуру в виде параметра, был описан пользовательский тип данных – массив вещественных элементов с индексами от 1 до n (n – константа).

Разработанные подпрограммы успешно выполняют заданную задачу:

* сумма отрицательных элементов: 15.8009;
* элементы, удовлетворяющие условию : -0.8;0.17;-0.0009.