**Условие задачи**

Разработать программу, выделив процедуры. Даны два целочисленных массива с положительными элементами: Х1, Х2, …, Хn и У1 У2, …, Уn. Если все элементы массива Х меньше всех целочисленных массива Y с соответствующими индексами, сформировать массив Z1, Z2, …, Zn по правилу: Zi = Yi! / (Xi!(Yi-Xi))!

**Первый шаг в решении задачи – анализ данных.**

Прежде всего необходимо учесть, чтобы наш алгоритм сооветсвовал нескольким важным факторам:

* Использование алгоритма было удобным;
* Алгоритм был простым и наглядным;

**Метод решения** сравнение двух целочисленных массивов и на основе результата сформировать выходной массив Z по определенному правилу.

**Состав данных**

|  |  |  |  |
| --- | --- | --- | --- |
| **Имя** | **Смысл** | **Тип** | **Структура** |
| **Исходные данные** | | | |
| *array1,array2* | одномерный массив | одномерный массив целых чисел | одномерный массив целых чисел |
| *temp\_n, n* | Временная переменная для хранения количества элементов в массиве | целый | переменная |
| **Выходные данные** | | | |
| *array3* | одномерный массив вещественных чисел Z, который необходимо сформировать по условию задачи | одномерный массив вещественных чисел | одномерный массив вещественных чисел |
| **Промежуточные данные** | | | |
| *x* | временные переменная, выделенная под хранения индекса или переиспользуется под переменную цикла | целый | переменная |

**Форма ввода**

n

<*n*>

**Форма вывода**

-----------------------------------------------------

array1[0], …,array1[n]

-----------------------------------------------------

array2[0], …, array2[n]

-----------------------------------------------------

<ПРОМЕЖУТОЧНЫЕ ВЫЧИСЛЕНИЯ ДЛЯ ДЕБАГА> [0]

…

<ПРОМЕЖУТОЧНЫЕ ВЫЧИСЛЕНИЯ ДЛЯ ДЕБАГА> [n]

-----------------------------------------------------

array3[0], …, array3[n]

**Листинг программы, написанной на языке object pascal и скомпилированной с помощью freepascal**

**program** lab\_4\_3\_3;

**uses** math, sysutils;

**type**

arraytype = **array of** longint;

arrayrealtype = **array of** currency;

**function** big\_brain\_formula(n: integer): longint;

**var**

factorial: longint;

i: longint;

**begin**

factorial := 1;

**for** i := 2 **to** n **do**

**begin**

factorial := factorial \* i;

**end**;

big\_brain\_formula := factorial;

**end**;

**procedure** big\_brain\_procedure(**var** c: arrayrealtype; a, b: arraytype);

**var**

i: longint;

**begin**

**for** i := 0 **to** length(a) - 1 **do**

**begin**

**if** a[i] > b[i] **then exit**;

**end**;

**for** i := 0 **to** length(a) - 1 **do**

**begin**

writeln('factorial ', i);

writeln('a[', i, '] ', a[i]);

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

writeln('chislitel ', big\_brain\_formula(b[i]));

writeln('znameneel ', (big\_brain\_formula(a[i]) \* big\_brain\_formula(b[i] - a[i])));

c[i] := (big\_brain\_formula(b[i]) \* 1.0) / (big\_brain\_formula(a[i]) \* big\_brain\_formula(b[i] - a[i]));

writeln('-----------------------------------------------------');

**end**;

**end**;

**var**

n, temp\_n, x: longint;

array1, array2: arraytype;

array3: arrayrealtype;

**begin**

writeln('n');

read(temp\_n);

n := temp\_n - 1;

setlength(array1, temp\_n);

setlength(array2, temp\_n);

setlength(array3, temp\_n);

**for** x := 0 **to** n **do**

**begin**

array1[x] := random(3);

array2[x] := random(4) + 6;

**end**;

writeln('-----------------------------------------------------');

**for** x := 0 **to** n **do**

**begin**

write(array1[x], ' ');

**end**;

writeln('-----------------------------------------------------');

**for** x := 0 **to** n **do**

**begin**

write(array2[x], ' ');

**end**;

writeln('-----------------------------------------------------');

big\_brain\_procedure(array3, array1, array2);

**for** x := 0 **to** n **do**

**begin**

writeln(array3[x]);

**end**;

writeln('-----------------------------------------------------');

**end**.