МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ   
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ**

**ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В. Г. ШУХОВА»**

**(БГТУ им. В.Г. Шухова)**

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

**Лабораторная работа №4**

по дисциплине: Основы программирования

тема: «Обработка одномерных массивов с использованием подпрограмм»

Выполнил: ст. группы ПВ-201

Машуров Дмитрий Русланович

Проверил:

Притчин Иван Сергеевич

Белгород 2020 г.

**Лабораторная работа №4**

**«Обработка одномерных массивов с использованием подпрограмм»**

**Цель работы**: получение навыков работы с массивами и подпрограммами.

**Задания для подготовки к работе:**

1. Изучите способы описания и использования массивов, базовые алгоритмы обработки массивов.
2. Изучите виды и назначение подпрограмм, правила их описания и вызова.
3. Разбейте задачу соответствующего варианта на подзадачи, таким образом, чтобы решение каждой подзадачи описывалось подпрограммой, а основная программа состояла бы в основном из вызовов подпрограмм.
4. Опишите блок-схему алгоритма решения задачи в укрупненных блоках.
5. Опишите используемые структуры данных, спецификации и блок-схемы подпрограмм, соответствующих укрупненным блокам. Спецификация содержит: заголовок подпрограммы, назначение, входные и выходные параметры.
6. Опишите блок-схему алгоритма решения задачи с использованием блоков «предопределенный процесс».
7. Закодируйте алгоритм.
8. Подберите наборы тестовых данных с обоснованием их выбора.

**Задания к работе:**

1. Наберите программу, отладьте ее и протестируйте.
2. Выполните анализ ошибок, выявленных при отладке программы.

**Задание варианта №17**

Даны два множества целых чисел. Выяснить, является ли одно из них подмножеством другого. Если является, то упорядочить подмножество

**Выполнение работы:**

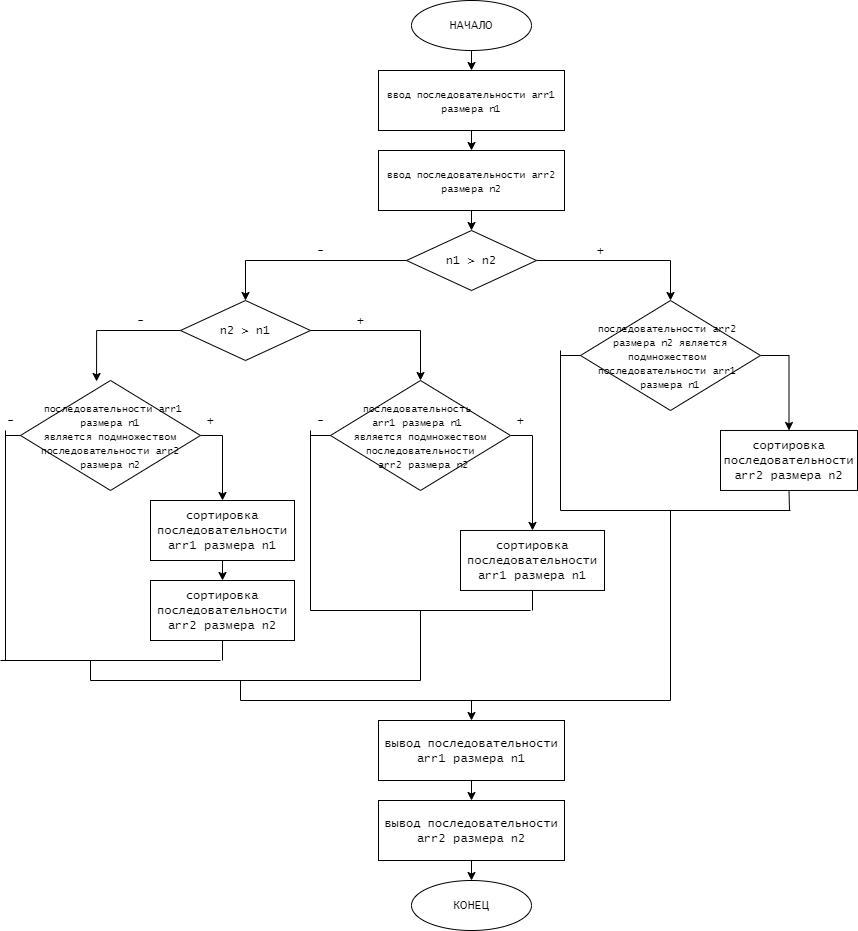
1. **Выделение подзадач**

Выделим следующие подзадачи:

1. Ввод первой последовательности arr1 размера n1
2. Ввод второй последовательности arr2 размера n2
3. Проверка на то, является ли одна последовательность подмножеством другой последовательности
4. Сортировка последовательности в порядке неубывания
5. Вывод первой последовательности arr1 размера n1
6. Вывод второй последовательности arr2 размера n2

Опишем алгоритм в укрупнённых блоках в терминах выделенных подзадач.

1. **Блок-схема алгоритма в укрупнённых блоках**

****

1. **Описание структур данных:**

SIZE – константа, определяющая максимальный размер последовательности, равный 100

t\_arr – тип описывающий массив размера SIZE

const

SIZE = 100;

type

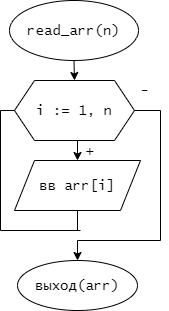
t\_arr = array [1..SIZE] of integer;

1. **Описание подпрограмм**

Спецификация процедуры read\_arr

1. Заголовок: procedure read\_arr(var arr: t\_arr; n: integer)
2. Назначение: ввод последовательности arr размера n
3. Входные параметры: n
4. Выходные параметры: arr

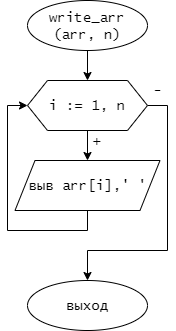
Блок-схема:



Спецификация процедуры write\_arr

1. Заголовок procedure write\_arr(arr: t\_arr; n: integer)
2. Назначение: вывод последовательности arr размера n
3. Входные параметры: arr, n
4. Выходные параметры: нет

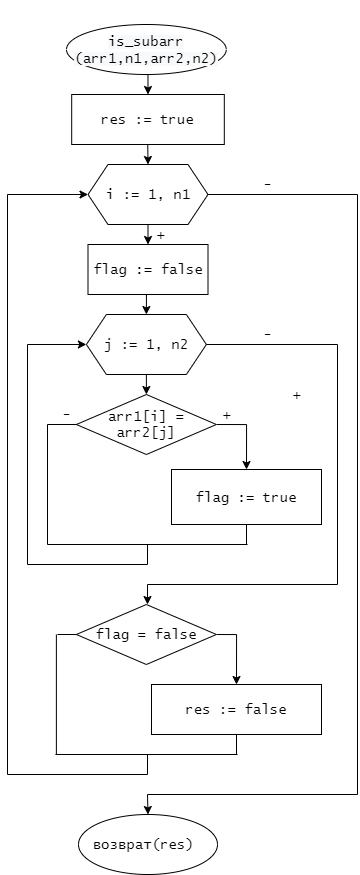
Блок-схема:



Спецификация функции is\_subarr

1. Заголовок function is\_subarr(arr1, n1, arr2, n2) : boolean
2. Назначение: возвращает значение “истина” если множество arr1 размера n1 является подмножеством arr2 размера n2 и значение “ложь”, если не является
3. Входные параметры: arr1, n1, arr2, n2
4. Выходные параметры: res

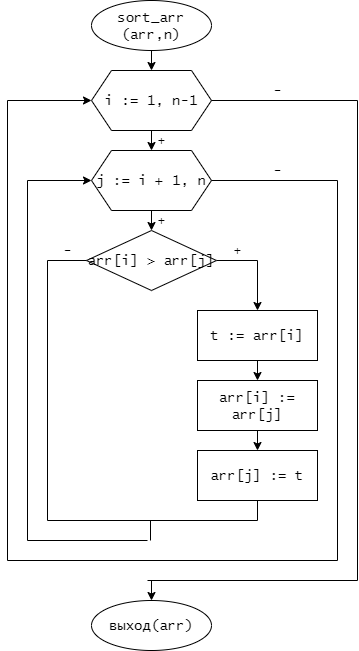
Блок-схема:



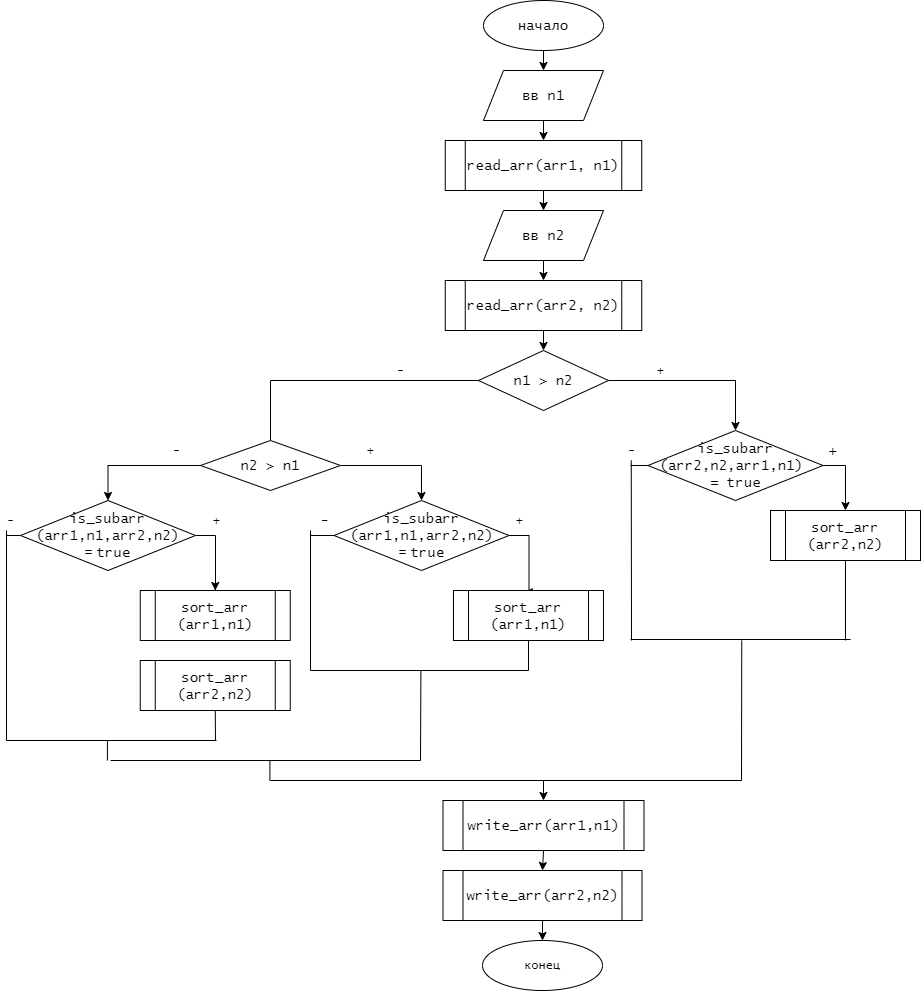
Спецификация процедуры sort\_arr

1. Заголовок procedure sort\_arr(var arr: t\_arr; n: integer)
2. Назначение: сортировка последовательности arr размера n в порядке неубывания
3. Входные параметры: arr, n
4. Выходные параметры: arr

Блок-схема:



1. **Блок-схема с блоками «предопределённый процесс»**

****

1. **Тестовые данные:**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| №  п/п | №  мн-ва | a1 | a2 | a3 | a4 | a5 | a6 | a7 | a8 | a9 | a10 | Результат |
| 1 | 1 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | Последовательность 2 является подмножеством последовательности 1 |
| 2 | 4 | 2 | 3 | 1 |  |  |  |  |  |  |
| 2 | 1 | 5 | 7 | 10 | 67 | 8 | 9 | 12 | 15 | 6 | 18 | Ни одна последовательность не является подмножеством другой |
| 2 | 5 | 5 | 18 | 12 |  |  |  |  |  |  |
| 3 | 1 | 10 | 228 | 2 | 4 | 10 |  |  |  |  |  | Последовательность 1 является подмножеством последовательности 2 |
| 2 | 10 | 7 | 10 | 8 | 9 | 2 | 4 | 5 | 228 | 0 |

1. **Текст программы:**

**const**

SIZE = 100;

**type**

t\_arr = **array** [1..SIZE] **of** integer;

{процедура ввода множества}

**procedure** read\_arr(**var** arr: t\_arr; n: integer);

**var**

i: integer;

**begin**

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

read(arr[i]);

**end**;

{процедура вывода множества}

**procedure** write\_arr(arr: t\_arr; n: integer);

**var**

i: integer;

**begin**

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

write(arr[i], ' ');

**end**;

{процедура проверки на принадлежность множества arr1 множеству arr2}

**function** is\_subarr(arr1: t\_arr; n1: integer; arr2: t\_arr; n2: integer) : boolean;

**var**

i,j: integer;

flag: boolean;

res: boolean;

**begin**

res := true;

**for** i := 1 **to** n1 **do**

**begin**

flag := false;

**for** j := 1 **to** n2 **do**

**if** (arr1[i] = arr2[j]) **then**

flag := true;

**if** (flag = false) **then**

res := false;

**end**;

is\_subarr := res;

**end**;

{процедура сортировки множества в порядке неубывания}

**procedure** sort\_arr(**var** arr: t\_arr; n: integer);

**var**

i, j: integer;

t: integer;

**begin**

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

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

**if** (arr[i] > arr[j]) **then**

**begin**

t := arr[i];

arr[i] := arr[j];

arr[j] := t;

**end**;

**end**;

**var**

arr1, arr2: t\_arr;

n1, n2: integer;

**begin**

write('Введите размер первого множества (размер не превышает 100): ');

read(n1);

writeln('Вводите значения первого множества');

read\_arr(arr1, n1);

write('Введите размер второго множества (размер не превышает 100): ');

read(n2);

writeln('Вводите значения второго множества');

read\_arr(arr2, n2);

**if** (n1 > n2) **then**

**begin**

**if** (is\_subarr(arr2, n2, arr1, n1) = true) **then**

sort\_arr(arr2, n2);

**end**

**else if** (n2 > n1) **then**

**begin**

**if** (is\_subarr(arr1, n1, arr2, n2) = true) **then**

sort\_arr(arr1, n1);

**end**

**else**

**begin**

**if** (is\_subarr(arr1, n1, arr2, n2) = true) **then**

**begin**

sort\_arr(arr1, n1);

sort\_arr(arr2, n2);

**end**;

**end**;

writeln('Вывод итоговых последовательностей');

writeln('Значения первого множества');

write\_arr(arr1, n1);

writeln(' ');

writeln('Значения второго множества');

write\_arr(arr2, n2);

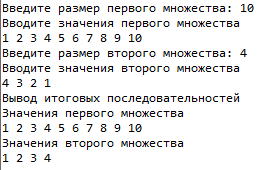
**end**.

1. **Анализ допущенных ошибок:**

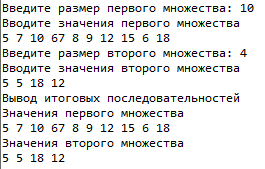
* В конце сортировал последовательность arr1 размера n2 вместо размера n1

1. **Результаты работы программы:**

*Пример №1:*



*Пример №2:*



Пример №3:

