Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Вятский государственный университет»

Колледж ВятГУ

**ОТЧЕТ**

**ПО ДОМАШНЕЙ КОНТРОЛЬНОЙ РАБОТЕ №2**

**«Вычисление значения функции»**

**ПО ДИСЦИПЛИНЕ «Основы алгоритмизации и программирования»**

Выполнил: студент учебной группы

ИСПк-205-52-00 Головёшкин Андрей Евгеньевич

Преподаватель:

Сергеева Елизавета Григорьевна

Киров

2024

**Вариант 2**

**Цель:** получить базовые навыки работы с одномерными массивами, освоить принципы работы со строками как с частным случаем одномерных массивов.

**Задание:**

1. Написать программу, выполняющую поиск двух элементов заданного массива, таких, что их разность по модулю минимальна.
2. Написать программу, определяющую является ли заданная строка подпоследовательностью исходной строки.

**Описание алгоритма:**

**Задание 1**

1. Вводим смысловые и технические переменные:
   * min\_diff: здесь будет храниться минимальная разность, MaxInt (максимальное возможное значение для целого числа
   * elem1 и elem2: Здесь хранятся элемента массива с минимальной разностью
2. Два вложенных цикла:
   * Первый цикл (i) проходит по всем элементам массива с индексами от 1 до 6 (включительно).

* Второй цикл (j) проходит по всем элементам массива с индексами, начиная с i + 1 до 6 включительно. Это исключает повторные проверки пар (например, пары (i, j) и (j, i)) Вычисление разности и обновление минимальной разности:
  + Внутри вложенных циклов для каждой пары элементов массива (lst[i] и lst[j]) вычисляется их абсолютная разность с помощью функции Abs(lst[i] - lst[j]).
  + Если эта разность меньше текущего значения min\_diff, то:
* Значение min\_diff обновляется на новое значение разности.
* Переменные elem1 и elem2 получают значения элементов массива lst[i] и lst[j], соответствующих минимальной разности.

**Схема алгоритма**

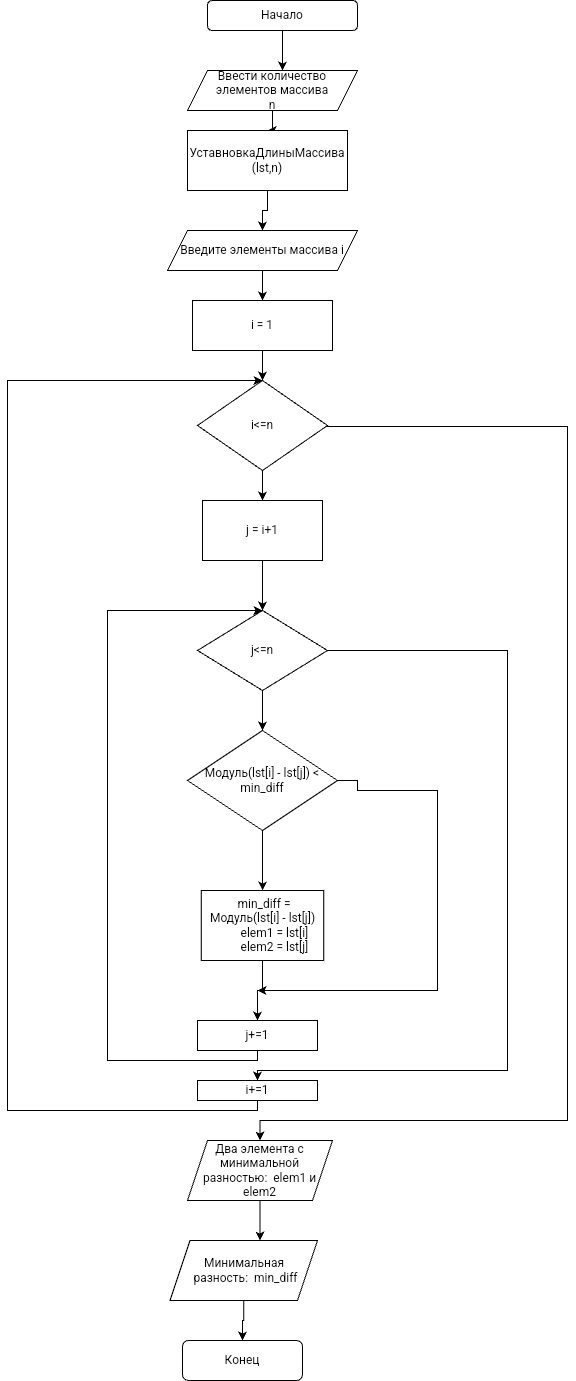


Схема задание 1

**Описание алгоритма:**

**Задание 2**

1. **Ввод данных:**

* Сначала программа запрашивает у пользователя ввод изначальной строки orig\_str.
* Затем программа запрашивает ввод подстроки subseq\_str, которую необходимо проверить на подпоследовательность.

1. **Проверка длины строк:**

* Если длина подстроки subseq\_str больше длины оригинальной строки orig\_str, то подстрока не может быть подпоследовательностью

1. **Основной цикл проверки:**

* Вводим индексы i,j для оригинальной строки и подстроки
* Алгоритм поочередно проверяет каждый символ оригинальной строки и пытается найти соответствующий символ в подстроке. Если символы совпадают, индекс j увеличивается на 1 (показывает на след.символ подстроки), а индекс i увеличивается на 1 (для перехода к следующему символу оригинальной строки).

1. **Завершение проверки:**

* Если все символы подстроки были найдены в оригинальной строке в том же порядке, то флаг is\_subseq останется равным True.
* Если хотя бы один символ подстроки не был найден в соответствующем порядке в оригинальной строке, то флаг is\_subseq устанавливается в False.

1. **Вывод результата:**

* В конце программа выводит сообщение:
  + "Строка является подпоследовательностью строки", если флаг is\_subseq равен True.
  + "Строка не является подпоследовательностью строки", если флаг is\_subseq равен False.

**Схема алгоритма**

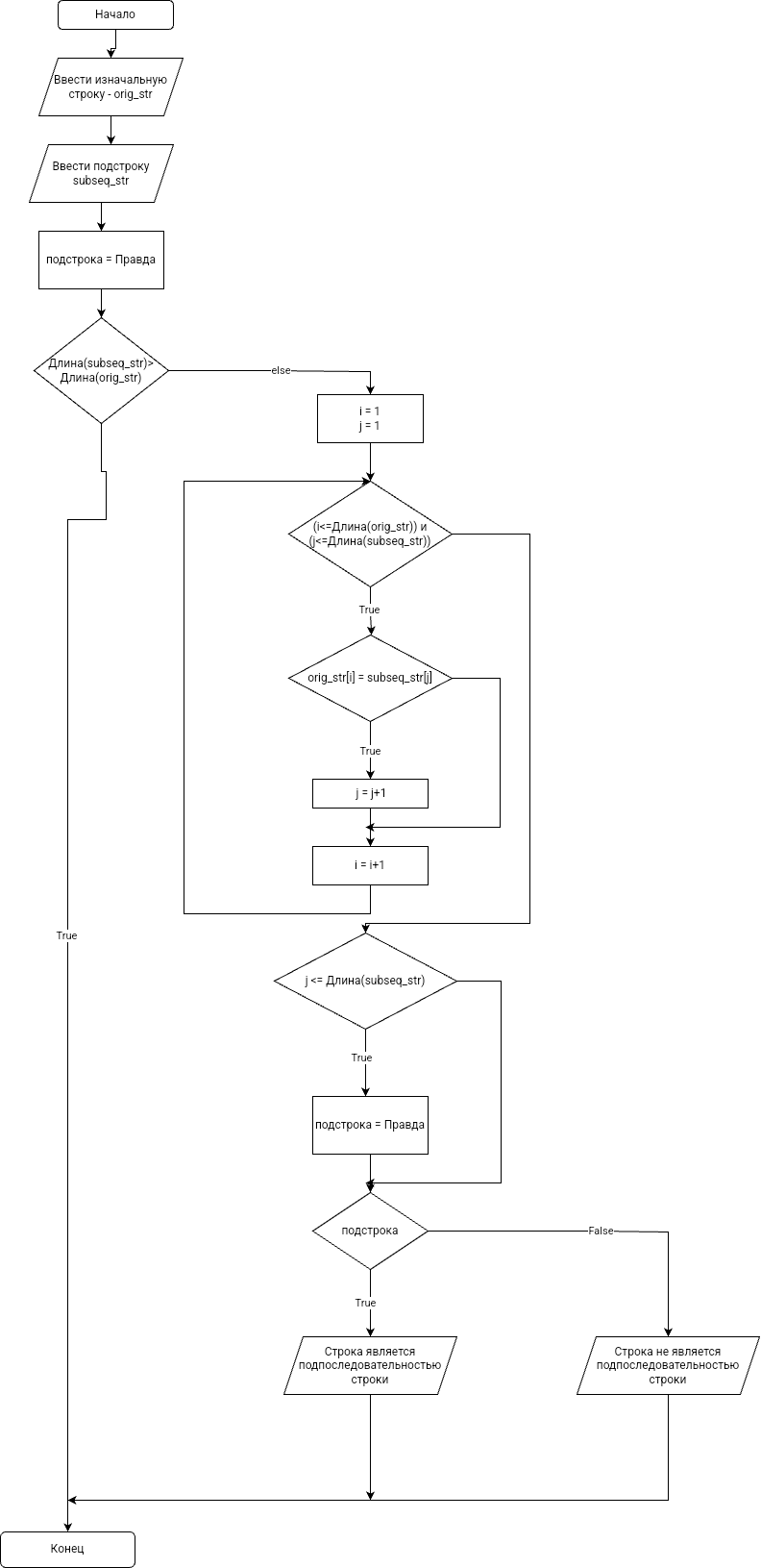


Схема задание 2

**Коды программ**

**1.**

program my52;

var

lst: array of Integer;

n, i, j: Integer;

min\_diff: Integer;

elem1, elem2: Integer;

begin

Write('Введите количество элементов массива: ');

ReadLn(n);

SetLength(lst, n);

WriteLn('Введите элементы массива:');

for i := 0 to n - 1 do

begin

Read(lst[i]);

end;

min\_diff := MaxInt;

elem1 := -1;

elem2 := -1;

for i := 0 to n - 1 do

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

begin

if Abs(lst[i] - lst[j]) < min\_diff then

begin

min\_diff := Abs(lst[i] - lst[j]);

elem1 := lst[i];

elem2 := lst[j];

end;

end;

WriteLn('Два элемента с минимальной разностью: ', elem1, ' и ', elem2);

WriteLn('Минимальная разность: ', min\_diff);

end.

**2.**

program my52;

var

orig\_str,subseq\_str:string;

i,j:integer;

is\_subseq:boolean;

begin

writeln('Введите изначальную строку');

readln(orig\_str);

writeln('Введите подстроку');

readln(subseq\_str);

is\_subseq := True;

if Length(subseq\_str)>Length(orig\_str) then

begin

is\_subseq:= False

end

else

begin

i:=1;

j:=1;

while (i<=Length(orig\_str)) and (j<=Length(subseq\_str)) do

begin

if orig\_str[i] = subseq\_str[j] then

j := j+1;

i := i+1;

end;

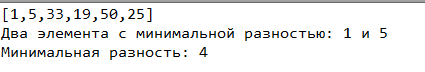
if j <= Length(subseq\_str) then

is\_subseq := False;

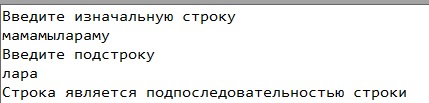
end;

**Результат выполнения программы**

**1.**

****

**2.**

****

**Вывод:**

**1 программа**

Эта программа находит пару элементов массива, разность между которыми минимальна. Для этого она перебирает все возможные пары элементов и находит минимальную разность. Программа работает корректно, выводя два элемента с минимальной разностью и саму минимальную разность.

**2 программа**

Эта программа проверяет, является ли подстрока подпоследовательностью оригинальной строки. Программа корректно выводит, является ли подстрока подпоследовательностью или нет.

**Заключение**

Обе программы работают корректно и демонстрируют правильные результаты. Первая программа находит два элемента с минимальной разностью в массиве, а вторая — проверяет, является ли одна строка подпоследовательностью другой. Обе задачи решаются эффективно, и программы продемонстрировали свою работоспособность на примерах.