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**

**«Изучение одномерных массивов и строк»**

**ПО ДИСЦИПЛИНЕ «Основы Алгоритмизации и Программирования»**

Выполнил: студент учебной группы

ИСПк-203-52-00

Яковлев Александр Станиславович

Преподаватель:

Сергеева Елизавета Григорьевна

Киров

2024

В отчете должны отображаться:

1. Цель работы
2. Формулировка задания (с вариантом)
3. Описание алгоритма и ответы на вопросы
4. Схема алгоритма с комментариями
5. Код программы
6. Результат выполнения программы
7. Вывод
8. **Цель работы** - получить базовые навыки работы с одномерными массивами, освоить принципы работы со строками как с частным случаем одномерных массивов.
9. **Формулировка задания:** Вариант 25.

**1 Задание**

Написать программу, выполняющую поиск такого максимального k, что сумма первых k элементов массива не превосходит заданного числа s.

**2 Задание**

Написать программу, удаляющую из строки каждое второе вхождение заданной подстроки.

**Описание алгоритма**

**1 Задание**

1. С клавиатуры вводится массив n;
2. Вводиться элементы массива, а также s;
3. Проверяется условие: если sum> s, программа заканчивается выводом k, в противном случае переменная k увеличивается на 1
4. Выводится k в качестве ответа.

**2 Задание**

1. Вводится с клавиатуры строка и подстрока;
2. Запускается цикл, проверяющий совпадает ли символ строки с первым символом подстроки;
3. Если символ строки совпал с первым символом подстроки, то строка и подстрока проверяются дальше;
4. Если они совпали, то счетчик k увеличивается на 1;
5. Если k четное, то счетчик i увеличивается на длину подстроки;
6. К результирующей переменной res прибавляется символ исходной строки с индексом i;
7. Выводится res в качестве ответа.
8. **Схема алгоритма с комментариями**

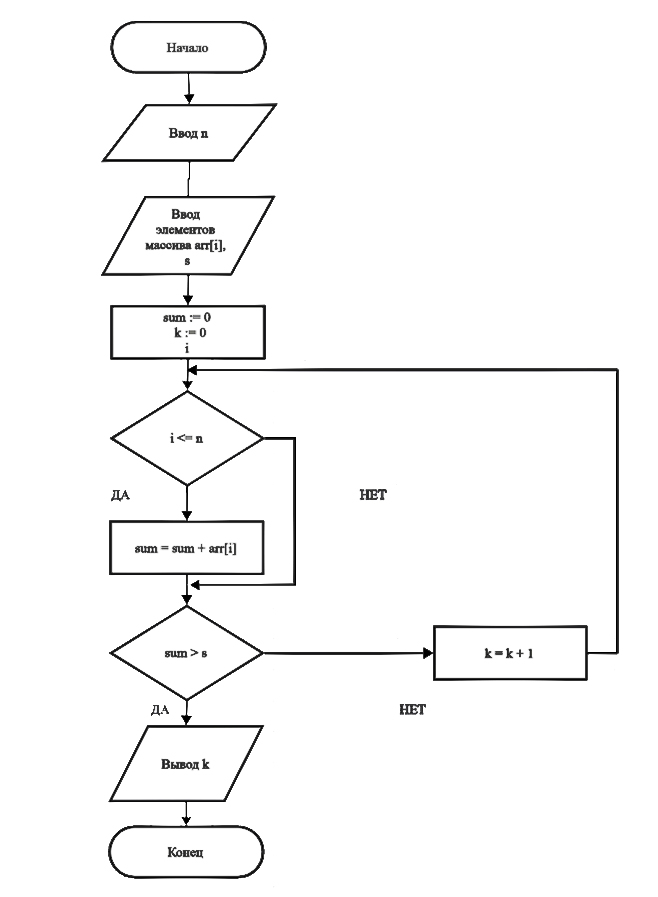


Рисунок 1 – Схема алгоритма для задания 1

Изображение выглядит как текст, диаграмма, чек, черно-белый

Автоматически созданное описание

Рисунок 2 – Схема алгоритма для задания 2

1. **Код программы**

**Задание 1.**

**program** FindMaxK;

**var**

arr: **array**[1..100] **of** integer;

n, s, i, sum, k: integer;

**begin**

write('Введите количество элементов в массиве: ');

ReadLn(n);

writeLn('Введите элементы массива:');

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

**begin**

read(arr[i]);

**end**;

write('Введите заданное число s: ');

readLn(s);

sum := 0;

k := 0;

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

**begin**

sum := sum + arr[i];

**if** sum > s **then**

**break**;

k := k + 1;

**end**;

writeLn('Максимальное значение k, для которого сумма первых k элементов не превосходит ', s, ': ', k);

**end**.

**Задание 2**.

**var** s, p, res: string;

g, k, w, len, i, i1: integer;

**begin**

writeln('введите строку');

readln(s);

writeln('введите подстроку');

readln(p);

k := 0;

i := 1;

len := length(p);

res := '';

**while** i <= length(s) **do**

**begin**

**if** (i <= length(s) - len + 1) **and** (s[i] = p[1]) **then**

**begin**

g := i;

w := 1;

**while** (w <= len) **and** (s[g] = p[w]) **do**

**begin**

g := g + 1;

w := w + 1;

**end**;

**if** w > len **then**

**begin**

k := k + 1;

**if** k **mod** 2 = 0 **then**

**begin**

i := i + len;

**continue**;

**end**;

**end**;

**end**;

res := res + s[i];

i := i + 1;

**end**;

writeln('результат: ', res);

**end**.

1. **Результат выполнения программы**

Рисунок 3.1 – 3.3- результаты выполнения программы – Задание 1

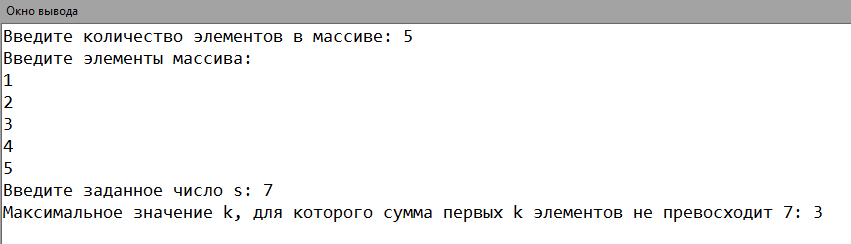


Рисунок 3.1 – Задание 1

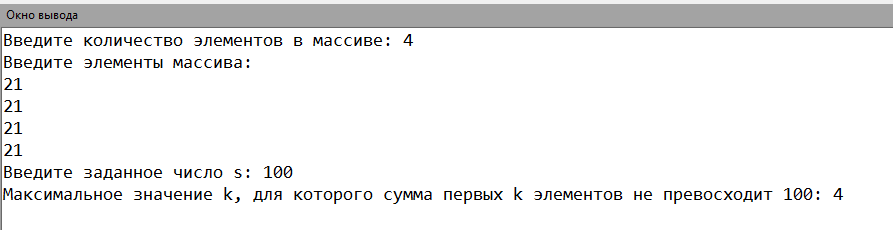


Рисунок 3.2 – Задание 1

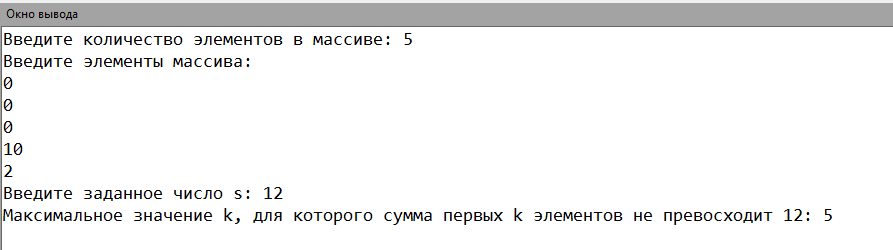


Рисунок 3.3 – Задание 1

Рисунок 3.4 – 3.6 - результаты выполнения программы – Задание 2

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание

Рисунок 3.4 – Задание 2

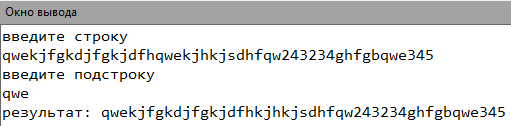


Рисунок 3.5 – Задание 2(2)

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 3.6 – Задание 2(3)

1. **Вывод:**

Освоение принципов работы со строками как с частным случаем одномерных массивов является еще одной важной целью при изучении программирования на Pascal и имеет ключевое значение для обработки текстовой информации. Строки в Pascal, представляющие собой последовательности символов, сохраняют все преимущества, которые предоставляет работа с одномерными массивами. Это значит, что программист, умеющий оперировать массивами, может легко применять эти навыки к строкам, выполняя широкий спектр операций, таких как поиск подстрок, конкатенация, замена символов и сортировка. Более того, работа со строками позволяет программистам эффективно обрабатывать текстовую информацию в приложениях, таких как обработка данных пользователя, работа с файлами и создание пользовательских интерфейсов. Понимание структуры строк и принципов их обработки не только улучшает качество программного кода, но и позволяет находить оптимальные решения для хранения и анализа текстовой информации, что является важным аспектом в современном программировании. В результате, освоение работы со строками расширяет инструментарий программиста и улучшает его способность решать разнообразные задачи в области разработки программного обеспечения.