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 №2

по дисциплине: Основы программирования
тема: «Стандартный ввод и вывод»

Выполнил: ст. группы ПВ-201
Машуров Дмитрий Русланович

Проверил:
Притчин Иван Сергеевич

Белгород 2021 г.

Лабораторная работа № 2

Основные понятия языка Си

Цель работы: знакомство с основными типами данных, операциями, операторами языка Си.

Задания для подготовки к работе:

1. Изучить базовые типы данных в языке Си и сравнить их с основными типами данных языка Паскаль.
2. Изучить арифметические операции и операции присваивания в Си.
3. Ознакомиться с операторами в Си.
4. Изучить стандартные математические функции библиотеки `math`.
5. Изучить правила описания функций и обращения к ним.
6. Разработать алгоритм и составить программу, состоящую, по крайней мере, из двух функций, для решения задачи соответствующего варианта. Результаты должны быть выведены в наиболее естественном виде. Например, если требуется многочлен $x^2 + 3x - 4$ разложить на множители, то результат должен быть выведен следующим образом: $x^2 + 3x - 4 = (x - 1)(x + 4)$
7. Подобрать тестовые данные

Задание варианта №17

Дана вещественная последовательность $a_1, a_2, \dots, a_n$. Определить максимальное количество идущих подряд положительных членов последовательности. Вывести найденный фрагмент.

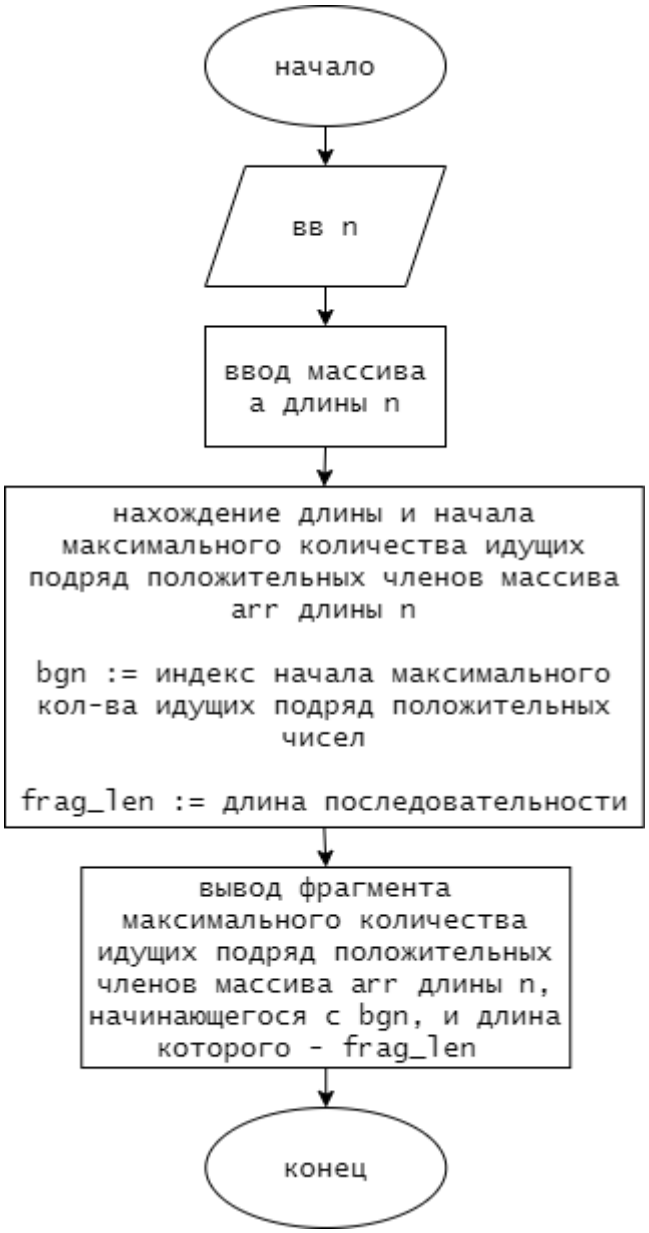
Выполнение:

1. **Описание алгоритма и выделение подзадач**
   Исходя из того, что нам нужно определить максимальное количество идущих подряд положительных членов последовательности, можно запоминать индекс начала фрагмента с максимальным количеством идущих подряд положительных членов последовательности и индекс конца.
   Затем выводим количество элементов: $frag\_len + 1$. А после сами элементы, которые располагаются между индексами bgn и $bgn + frag\_len$.

   Выделим следующие подзадачи:
   - Ввод последовательности
   - Нахождение максимального количества идущих подряд положительных членов последовательности
   - Вывод максимального количества идущих подряд положительных членов последовательности

   Далее алгоритм описан в укрупнённых блок в терминах выделенных подзадач

2. Блок-схема с укрупнёнными блоками



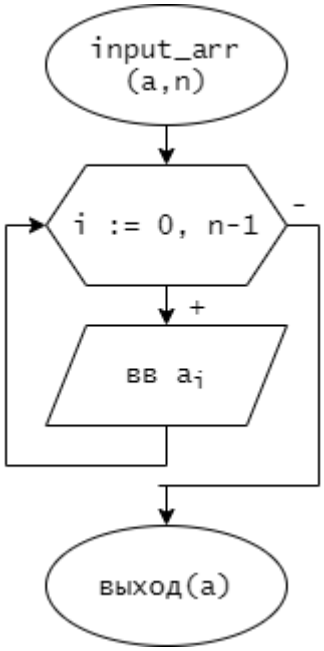
3. Спецификации подпрограмм

Ввод последовательности

Спецификация процедуры `input_arr`

1) Заголовок: `void input_arr(float a[], const size_t n)`
2) Назначение: ввод вещественного массива `a` длины `n`

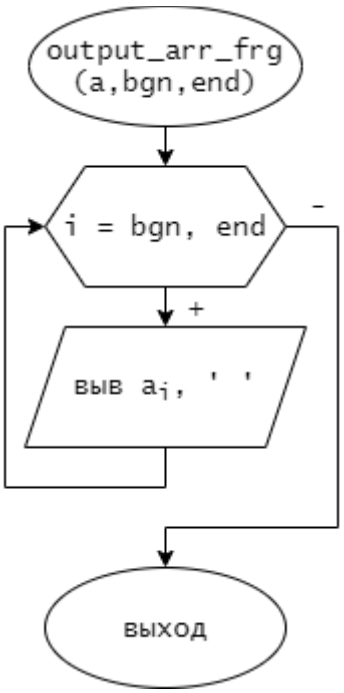
Блок-схема:



Вывод максимального количества идущих подряд членов последовательности

Спецификация процедуры `output_arr_frg`

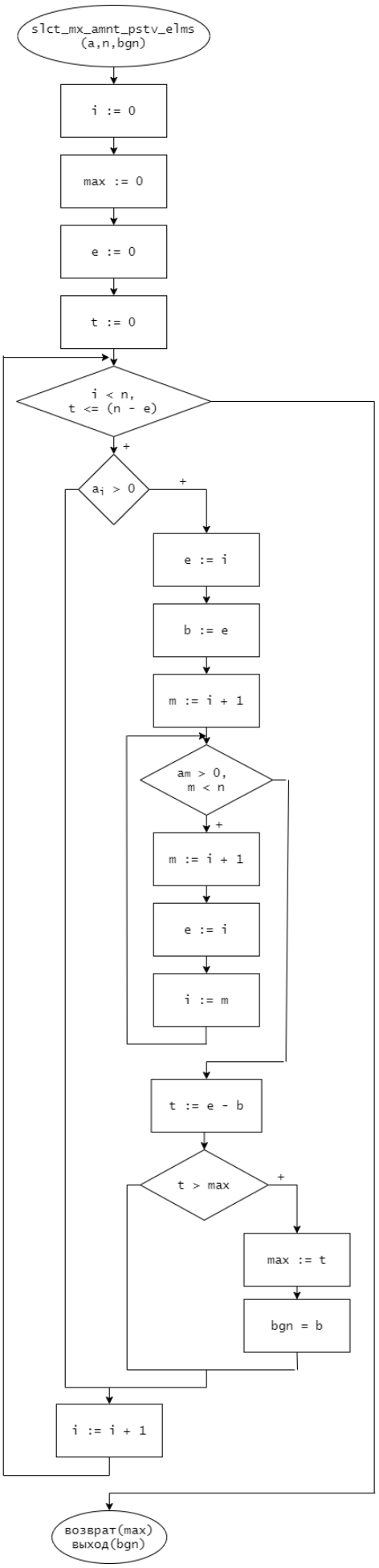
1) Заголовок: `void output_arr_frg(const float a[], const size_t bgn, const size_t end)`
2) Назначение: вывод фрагмента массива `a` с индекса `bgn` по индекс `end`
   Блок-схема:



Нахождение максимального количества идущих подряд положительных членов последовательности

Спецификация процедуры `get_len_mx_amnt_pstv_elms`

1) Заголовок: `get_len_mx_amnt_pstv_elms(const float a[], const size_t n, size_t bgn)`
2) Назначение: возвращает длину максимального количества идущих подряд положительных членов массива `a` длины `n`. Побочный эффект: в `bgn` помещается индекс начала фрагмента. Блок-схема:


slct_mx_amnt_pstv_elms
(a,n,bgn)
i := 0
max := 0
e := 0
t := 0
i < n,
t <= (n - e)
+
ai > 0
+
e := i
b := e
m := i + 1
am > 0,
m < n
+
m := i + 1
e := i
i := m
t := e - b
t > max
+
max := t
bgn = b
i := i + 1
возврат(max)
выход(bgn)

4. Тестовые данные

№	Вход	Выход
1	9 1.0 -2.22 3.4 4.1 5.01 –12.0 -32.02 9.9 10.1	3.4 4.1 5.0
2	6 -12.22 23.3 -1.1 13.37 2.28 14.48	13.4 2.28 14.48
3	5 2.3 -1.1 -23.6 -66.6 -5.5	2.3

5. Текст программы:

```
#include <stdio.h>

///ввод массива arr длины n.
void input arr(float a[], const size t n)
{
    for (size_t i = 0; i < n; ++i)
        scanf("%f", &a[i]);
}

///вывод проивзольного количества идущих подряд элементов
///массива arr длины n начинающегося с bgn и
///оканчивающегося на end.
void output_arr_frg(const float a[], const size_t bgn, const size_t
end)
{
    for (size_t i = bgn; i <= end; i++)
    {
        printf("%f ",a[i]);
    }
}

///возвращает длину максимального количества идущих подряд
///положительных членов массива a длины n.
///Побочный эффект: в bgn помещается индекс начала фрагмента.
size_t get_len_mx_amnt_pstv_elms(const float a[], const size_t n,
size_t *bgn)
{
    size_t i = 0,
           b,
           e = 0,
           t = 0,
           max = 0;

    while ((i < n) && (t <= (n - e)))
    {
        if (a[i] > 0)
        {
            b = e = i;

            while ((a[i+1] > 0) && (i + 1 < n))
                e = ++i;

            t = e - b;

            if (t >= max)
            {
                max = t;
                *bgn = b;
```

```
            }
        }

        i++;
    }

    return max;
}

int main()
{
    printf("Input length of numbers sequence\n");
    size_t n;
    scanf("%ud", &n);

    float a[n];
    printf("Input elements of numbers sequence\n");
    input_arr(a, n);

    size_t bgn = 0,
           frag_len;

    frag_len = get_len_mx_amnt_pstv_elms(a, n, &bgn);

    if ((bgn == 0) && (frag_len == 0) && (a[0] < 0))
    {
        printf("There is no such fragment");
    }
    else
    {
        printf("Length of fragment is %u\n", frag_len+1);
        output_arr_frg(a, bgn, bgn + frag_len);
    }
}
```

6. Результаты работы программы

Пример №1

```
Input length of numbers sequence
9
Input elements of numbers sequence
1.0 -2.22 3.4 4.1 5.01 -12.0 -32.02 9.9 10.1
3.4 4.1 5.0
```

Пример №2

```
Input length of numbers sequence
6
Input elements of numbers sequence
-12.22 23.3 -1.1 13.37 2.28 14.48
13.4 2.3 14.5
```

Пример №3

```
Input length of numbers sequence
5
Input elements of numbers sequence
2.3 -1.1 -23.6 -66.6 -5.5
2.3
```

7. **Анализ допущенных ошибок**
    - Выход за пределы массива из-за чего происходил захват «мусора» при выделении фрагмента