МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №1

по дисциплине «Программирование»

Тема: Условия, циклы, оператор switch

Студент гр. 0382		Тюленев Т. В.
Преподаватель		Жангиров Т. Р.

Санкт-Петербург
2020

Цель работы.

Ознакомиться с базовыми синтаксическими конструкциями языка Си.

Задание.

Вариант №1

Напишите программу, выделив каждую подзадачу в отдельную функцию.

Реализуйте программу, на вход которой подается одно из значений 0, 1, 2, 3 и массив целых чисел размера не больше 20. Числа разделены пробелами. Строка заканчивается символом перевода строки.

В зависимости от значения, функция должна выводить следующее:

0 : индекс первого отрицательного элемента. (*index_first_negative*)

1 : индекс последнего отрицательного элемента. (*index_last_negative*)

2 : Найти произведение элементов массива, расположенных от первого отрицательного элемента (включая элемент) и до последнего отрицательного (не включая элемент). (*multi_between_negative*)

3 : Найти произведение элементов массива, расположенных до первого отрицательного элемента (не включая элемент) и после последнего отрицательного (включая элемент). (*multi_before_and_after_negative*)

иначе необходимо вывести строку "Данные некорректны".

Основные теоретические положения.

В программе использовались следующие конструкции:

- Стандартная библиотека Си *studio.h;*
- Функции: *index_first_negative, index_last_negative, multi_between_negative, multi_before_and_after_negative u main;*
- Для вывода и ввода данных использовались функции: *Getchar, scanf u printf*;
- Также для хранения данных в программе были использованы переменные.
- Логические конструкции: *Switch, if, break;*
- Циклы: *for;*

Выполнение работы.

1. Функция *index_first_negative* **.**

Цель данной функции найти из стандартного ввода первое отрицательное число и вывести на экран его индекс. Сначала объявляются переменные *a, i* и *index_a*. Переменная *i* будет использоваться для цикла и обозначения, *index_a*. Далее используется цикл *For (*который завершает ввод данных при встрече символа переноса строки или после 20 элемента), где поочерёдно вводятся цифры, ищется индекс первого отрицательного числа и выводится на экран.

2. Функция *index_last_negative* **.**

Цель данной функции найти из стандартного ввода последнее отрицательное число и вывести на экран его индекс. Сначала объявляются переменные *a, i* и *index_a*. Переменная *i* будет использоваться для цикла и обозначения, *index_a*. Далее используется цикл *For* (который завершает ввод данных при встрече символа переноса строки или после 20 элемента), где поочерёдно вводятся цифры, ищется индекс последнего отрицательного числа и выводится на экран.

3. Функция *multi_between_negative* **.**

Цель данной функции найти произведение элементов, расположенных от первого отрицательного элемента (включая элемент) и до последнего отрицательного (не включая элемент) и вывести на экран. Сначала объявляются переменные *a, pointer = 0 ,min = 0 , max = 1.* Переменная *i* будет использоваться для цикла. При встрече первого отрицательного числа мы начинаем max умножать на вводимое число, а при встрече следующего отрицательного, мы приравниваем *min=max*. В конце функции мы выводим min, что является произведением элементов между первым (включая) и последним (не включая) отрицательным числом.

4. Функция *multi_before_and_after_negative*.

Цель данной функции найти произведение элементов, расположенных до первого отрицательного элемента (не включая) и до последнего отрицательного (включая) и вывести на экран. Сначала объявляются переменные *a, pointer = 0 ,min = 0 , max = 1.* Переменная *i* будет использоваться для цикла. До встречи первого отрицательного числа мы max умножаем на вводимое число, а при встрече отрицательного перестаём max умножать на вводимое число до встречи с последним отрицательным. В конце функции мы выводим min, что является произведением элементов до первого (не включая) и после последнего (включая) отрицательного числа.

5. Подключение стандартной библиотеки.

В начале файла подключается заголовочный файл *stdio.h,* являющийся стандартным заголовочным файлом языка Си.

6. Функция *main*.

Выполнение программы начинается с функции *main.* В начале программы объявляется несколько переменных и целочисленный массив. Затем пользователю предлагается ввести число от 0 до 3 (результат сохраняется в переменную `n`). Следующим шагом объявляется оператор *switch*, в котором будет отслеживаться значение переменной `n`. В каждом логическом блоке *case* вызывается нужная функция. Если же переменная не равняется ни одному из допустимых значений, то на экран выводится сообщение о том, что данные некорректны.

Тестирование.

Результаты тестирования представлены в табл. 1.

Таблица №1 – Результаты тестирования

№ п/п	Входные данные	Выходные данные	Комментарии
1	0 -5 -3 -5 -8 3 -9 -3	0	true
2	0 1 -2 3 4 5 6	1	true
3	0 1 2 -3 -4 -5 -6	2	true
4	0 1 2 3 4 5 6		true
5	1 1 2 3 4 -5 6	4	true
6	1 1 2 -3 4 -5 6	4	true
7	1 1 2 3 4 5 6		true
8	2 1 -2 3 4 -5 6	-24	true
9	2 1 -2 -3 -4 -5 6	-24	true
10	2 1 2 3 4 5 6	0	true
11	2 1 -2 3 4 5 6	-2	true
12	3 1 -2 3 4 -5 6	-30	true
13	3 1 -2 -3 -4 -5 6	-30	true
14	3 1 2 3 4 5 6	720	true
15	3 1 -2 3 4 5 6	-720	true
16	13 0 0 0 0 0 0 0	Данные некорректны	true

Выводы.

Удалось успешно применить и, тем самым закрепить знания базовых синтаксических конструкций языка Си.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

Исходный код программы хранится в .../src/main.c

```
#include <stdio.h>


void
index_first_negative()
{
 int a, index_a;
 index_a=-1;
  for  (int  i=0;  i<20;
i++){
  scanf("%d", &a);
  if (a<0){
  index_a=i;
  }
         if    ((a<0)||
(getchar()=='\n')){
  break;
 }
}
if (index_a>-1){
printf("%d\n",
index_a);
}
}


void
index_last_negative(){
 int a, index_a;
 index_a=-1;
  for  (int  i=0;  i<20;
i++){
  scanf("%i", &a);
```

```
  if (a<0){
   index_a=i;
  }
   if (getchar()=='\n')
{
   break;
  }
}
if (index_a>-1){
printf("%d\n",
index_a);
}
}


void
multi_between_negative
(){
   int  a,  pointer  =
0 ,min = 0 , max = 1;
  for  (int  i=0;  i<20;
i++){
  scanf("%i", &a);
  if (a<0){
   if (pointer==0){
    min=a;
   }
   else{
    min=max;
   }
  pointer=-1;
  }
  if (pointer==-1){
   max=max*a;
  }
   if (getchar()=='\n')
```

```
{
   break;
  }
}
printf("%i\n", min);
}


void
multi_before_and_after
_negative(){
   int   a,   pointer   =
0 ,min = 1 , max = 1;
  for  (int  i=0;  i<20;
i++){
  scanf("%i", &a);
  max=max*a;
  if(a<0){
   pointer=-1;
   max=a;
  }
  if (pointer==0){
   min=min*a;
  }
   if (getchar()=='\n')
{
   break;
  }
}
if (pointer==-1){
 min=min*max;
}
printf("%i\n", min);
}
int main()
```

```
{
 int n;
 scanf("%d", &n);
 switch(n){
  case 0:
    index_first_negativ
e();
  break;
  case 1:
    index_last_negative
();
  break;
  case 2:
    multi_between_negat
ive();
  break;
  case 3:
    multi_before_and_af
ter_negative();
  break;
  default:
       printf(  "Данные
некорректны\n" );
 }
 return 0;
 }
```