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

Студентка гр. 0382 Охотникова Г.С.

Преподаватель Жангиров Т.Р.

Санкт-Петербург
2020

Цель работы.

Научиться разбивать программу на функции, пользоваться оператором switch и работать с массивами в языке С.

Задание.

Вариант 6:

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 100. 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 сумму модулей элементов массива, расположенных от первого отрицательного элемента (включая элемент) и до последнего отрицательного (не включая элемент). (sum_between_negative)
3 : Найти сумму модулей элементов массива, расположенных до первого отрицательного элемента (не включая элемент) и после последнего отрицательного (включая элемент). (sum_before_and_after_negative)

иначе необходимо вывести строку "Данные некорректны".

Основные теоретические положения.

Оператор множественного выбора *switch*:

```
switch (выражение) {
   case константное выражение: операторы
   case константное выражение: операторы
   default: операторы
}
```

Выполнение работы.

length — счетчик длины массива, передается в функции.

a[C] — массив, который подается на вход.

a_size — размер массива.

sym — символ, обозначающий пробел.

res1, res2 — переменные, которым присваивается значение, которое возвращают функции *index_first_negative* и *index_last_negative*.

n — переменная, обозначающая выбор команды(0. 1. 2 или 3).

#define C 100 — макрос, задающий максимальный размер массива.

Функция *index_first_negative(int a[C], int length)* принимает на вход массив *a*, максимальный размер которого *C*, и длину введенного массива *length*. Переменная *i = -1* является обозначением индекса первого отрицательного элемента, также она нужна для сравнения индекса элемента в массиве, так как нужно найти индекс первого отрицательного элемента, в каждой итерации цикла *for* идет проверка значения *i*. Если *i = -1*, то цикл продолжается. В обратном случае, нужный индекс найден. Функция возвращает найденное значение.

Функция *index_last_negative(int a[C], int length)* принимает на вход массив a, максимальный размер которого *C*, и длину введенного массива *length*. Переменная *i1* будет обозначать индекс последнего отрицательного элемента. В цикле *for* идет проверка каждого элемента массива на отрицательность. Таким образом, функция вернет индекс последнего отрицательного элемента в массиве.

Функция *sum_between_negative(int a[C], int length)* принимает на вход массив a, максимальный размер которого C, и длину введенного массива *length*. Переменная *sum* — это искомая сумма. Переменные *i1* и *i2* обозначают индексы первого отрицательного элемента и последнего соответственно. Для получения этих значений используется вызов первых двух функций. Затем с помощью цикла *for* считается сумма модулей(функция *abs*) элементов от первого отрицательного элемента до последнего.

Функция *sum_before_and_after_negative(int a[C], int length)* принимает на вход массив *a*, максимальный размер которого *C*, и длину введенного массива *length*. Переменная *sum1* — это сумма модулей элементов до первого отрицательного, а переменная *sum2* — это сумма модулей элементов после последнего отрицательного. Переменные *i1* и *i2* обозначают индексы первого отрицательного элемента и последнего соответственно. Для получения этих значений используется вызов первых двух функций. Функция разбита на два цикла *for*, в которых отдельно считаются суммы модулей до первого отрицательного элемента и после последнего. Функция выводит сумму переменных *sum1* и *sum2*.

В функции *main* сначала на вход поступает выбранная команда(0, 1, 2 или 3), затем массив. В зависимости от выбранной команды оператор множественного выбора вызывает соответствующую команде функцию и выводит результат. Если выбранной команды не существует, выводится сообщение «Данные некорректны».

Тестирование.

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

№ п/п	Входные данные	Выходные данные	Комментарии
1.	0 1 16 2 -18 -22 15 -3 13 0 -6 1 9 24 1 -18 15 28 20 -17 16 -11	3	Индекс первого отрицательного элемента равен 3.
2.	1 1 16 2 -18 -22 15 -3 13 0 -6 1 9 24 1 -18 15 28 20 -17 16 -11	20	Индекс последнего отрицательного элемента равен 20.
3.	2 1 16 2 -18 -22 15 -3 13 0 -6 1 9 24 1 -18 15 28 20 -17 16 -11	226	Сумма модулей элементов между первым отрицательным элементом(включая) и последним отрицательным(не включая) равна 226.
4.	3 1 16 2 -18 -22 15 -3 13 0 -6 1 9 24 1 -18 15 28 20 -17 16 -11	30	Сумма модулей элементов до первого отрицательного(не включая) и после последнего отрицательного(включая) равна 30.
5.	5 4 5 7 9 -2 4	Данные некорректны	Введена несуществующая команда. Выведено сообщение об ошибке.
6.	2 23 3 4 5 0 0 -8 -11 4 5 78 34 0 0	8	Сумма модулей элементов между первым отрицательным элементом(включая) и последним отрицательным(не включая) равна 8.

Выводы.

Было изучено, как разбивать программу на функции для того, чтобы сделать ее более лаконичной, и как работать с оператором множественного выбора *switch.*

Разработана программа, выполняющая считывание с клавиатуры исходных данных и команды пользователя. Для выбора команды использовался оператор множественного выбора switch, а для обработки команд в функциях использовались условные операторы *if-else* и циклы *for*.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: labr1.c

```
#include <stdio.h>
#include <stdlib.h>
#define C 100

int index_first_negative(int a[C], int lenght) {
	int r = -1;
	for (int i = 0; i < lenght; i++) {
		if (a[i] < 0 && r == -1) {
			r = i;
		}
	}
	return r;
}

int index_last_negative(int a[C], int lenght) {
	int i1;
	for (int i = 0; i < lenght; i++) {
		if (a[i] < 0) {
			i1 = i;
		}
	}
	return i1;
}


int sum_between_negative(int a[C], int lenght) {
	int sum = 0;
	int i1, i2, i;
	i1 = index_first_negative(a, lenght);
	i2 = index_last_negative(a, lenght);
	for (i = i1; i < i2; i++) {
		sum = sum + abs(a[i]);
	}
	printf("%d", sum);


}


int sum_before_and_after_negative(int a[C], int lenght) {
	int sum1 = 0, sum2 = 0;
	int i;
	int i1, i2;
	i1 = index_first_negative(a, lenght);
	i2 = index_last_negative(a, lenght);
	for (i = 0; i < i1; i++) {
		sum1 = sum1 + abs(a[i]);
	}
	for (i = i2; i < lenght; i++) {
		sum2 = sum2 + abs(a[i]);
```

```
	}
	printf("%d", (sum1 + sum2));
}


int main()
{
	int lenght = 0;
	int a[C];
	int a_size = 0;
	char sym = ' ';
	int res1, res2, n;
	scanf("%d", &n);
	while (a_size < C && sym == ' ') {
		scanf("%d%c", &a[a_size++], &sym);
		lenght++;
	}




	switch (n) {

	case 0:
		res1 = index_first_negative(a, lenght);
		printf("%d", res1);

		break;
	case 1:
		res2 = index_last_negative(a, lenght);
		printf("%d", res2);
		break;
	case 2:
		sum_between_negative(a, lenght);

		break;
	case 3:
		sum_before_and_after_negative(a, lenght);

		break;
	default:
		printf("Данные некорректны");
		return 0;
	}

	return 0;
}
```