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. 1304 _____ Крупин Н. С.

Преподаватель _____ Чайка К. В.

Санкт-Петербург

2021

Цель работы.

Усвоить на практике использование в языке С условий, циклов, а также оператора выбора switch.

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".

Ошибкой в данном задании считается дублирование кода!

Подсказка: функция нахождения модуля числа находится в заголовочном файле stdlib.h стандартной библиотеки языка Си.

При выводе результата не забудьте символ переноса строки».

Выполнение работы.

Фукция main начинается с последовательного считывания входных данных: код задачи от 0 до 3 перехватывается переменной type; статически выделяется память на 100 целочисленных значений, указатель на её начало хранится в переменной arr; далее адрес начала массива передаётся в функцию scan_arr, которая считывает массив и записывает по переданному адресу, возвращая количество задействованных элементов, которое перехватывается переменной count. Далее реализуется вывод результата в зависимости от кода задачи, используется оператор switch, для каждого случая печатается результат работы одной из функций, также проводится примитивная обработка ошибок – в случае ввода несуществующего кода задачи или попытки вывода индекса не присутствующего в массиве отрицательного элемента вызывается функция print_err, печатающая сообщение об ошибке.

Функция scan_arr принимает в качестве аргумента указатель на начало массива arr и записывает в него числа из введённой строки, пока не встретит символ её окончания. Функция использует переменную-счётчик i для хранения текущего индекса элемента и накапливания их количества и символьную переменную gap, хранящую в себе символ, следующий за введённым числом, для возможности его проверки. Функция возвращает количество записанных элементов.

Функция print_err печатает на экран сообщение «Данные некорректны», ничего не возвращает.

Функция index_first_negative принимает в качестве аргументов указатель на начало массива arr и количество его элементов count. Использует переменную-счётчик i для хранения текущего индекса элемента и сохранения после выполнения цикла индекса первого отрицательного элемента, а если такового не окажется, принимает значение count. Функция возвращает индекс первого отрицательного элемента или значение count, если такого не существует.

Функция index_last_negative принимает в качестве аргументов указатель на начало массива arr и количество его элементов count. Использует переменную-счётчик i для хранения текущего индекса элемента и сохранения после выполнения цикла индекса последнего отрицательного элемента, а если такового не окажется, принимает значение -1. Функция возвращает индекс последнего отрицательного элемента или значение -1, если такого не существует.

Функция sum_between_negative принимает в качестве аргументов указатель на начало массива arr и количество его элементов count. Использует переменную-счётчик i для хранения текущего индекса элемента и переменную sum для накопления суммы модулей элементов с индексами от index_first_negative (включительно) до index_last_negative (не включительно), данные значения индексов подсчитываются названными функциями. Функция возвращает сумму модулей элементов в указанном промежутке или значение 0, если промежуток пуст (при единственном отрицательном или их полном отсутствии).

Функция sum_before_and_after_negative принимает в качестве аргументов указатель на начало массива arr и количество его элементов count. Использует переменную-счётчик i для хранения текущего индекса элемента и переменную sum для накопления суммы модулей элементов с индексами меньше index_first_negative (не включительно) либо больше index_last_negative (включительно), данные значения индексов подсчитываются названными функциями. Функция возвращает сумму модулей элементов в указанных промежутках, а в случае единственного отрицательного элемента или их полном отсутствии – сумму модулей всех элементов массива.

Разработанный программный код см. в приложении А.

Тестирование.

Результаты тестирования представлены в таблице 1. Все результаты соответствуют ожидаемым.

Таблица 1 – Результаты тестирования

№ п/п	Входные данные	Выходные данные	Комментарии
1.	0 1 -2 3 -4 5 -6 7 8	1	Больше 2 отрицательных.
2.	1 1 -2 3 -4 5 -6 7 8	5	
3.	2 1 -2 3 -4 5 -6 7 8	14	
4.	3 1 -2 3 -4 5 -6 7 8	22	
5.	0 -2 3 -4	0	2 отрицательных.
6.	1 -2 3 -4	2	
7.	2 -2 3 -4	5	
8.	3 -2 3 -4	4	
9.	0 1 -2 3	1	1 отрицательный
10.	1 1 -2 3	1	
11.	2 1 -2 3	0	
12.	3 1 -2 3	6	
13.	0 0 1 2	Данные некорректны	Нет отрицательных.
14.	1 0 1 2	Данные некорректны	
15.	2 0 1 2	0	
16.	3 0 1 2	3	
17.	-1 0 -1 2 -3 4	Данные некорректны	Неверный код задачи.
18.	4 0 -1 2 -3 4	Данные некорректны	
19.	0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 -100	99	Массив на 100.

Выводы.

Были изучены основные управляющие конструкции языка C, а именно условия, циклы и оператор switch.

Разработана программа, выполняющая считывание с клавиатуры исходных данных и команды пользователя (кода задачи). Для обработки команд пользователя использовался оператор switch, позволяющий также выполнить примитивную проверку корректности введённого кода. Для поиска соответствующих значений в вычисляющих функциях применялись циклы for и условные операторы if. Во избежание возникновения исключительных ситуаций при выводе индекса несуществующего элемента в главной функции добавляется обработка оператором if-else.

Не проведена полная проверка на соответствие введённых данных заявленным требованиям (такая задача не формулировалась, но программа из-за этого становится опасной в применении – например, может обращаться к невыделенной памяти, если будет введено более 100 чисел массива).

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: Krupin_Nikita_lb1.c

```c
#include <stdio.h>
#include <stdlib.h> //int abs(int);
#define LIMIT 100 //Limit of array.

int index_first_negative(int* arr, int count){
  int i;
  for (i = 0; i < count; i++)
    if (arr[i] < 0) break;
  return i;
  //If i = count, there aren't negatives.
}

int index_last_negative(int* arr, int count){
  int i;
  for (i = count-1; i >= 0; i--)
    if (arr[i] < 0) break;
  return i;
  //If i < 0, there aren't negatives.
}

int sum_between_negative(int* arr, int count){
  int sum = 0;
  for (int i = index_first_negative(arr, count);
                          i < index_last_negative(arr, count); i++)
    sum += abs(arr[i]);
  return sum;
  //If there aren't negatives or there's only one, sum = 0.
}

int sum_before_and_after_negative(int* arr, int count){
  int sum = 0;
  for (int i = 0; i < count; i++)
    if (i < index_first_negative(arr, count) || i >=
                                  index_last_negative(arr, count))
      sum += abs(arr[i]);
  return sum;
  //If there aren't negatives or there's only one, return sum of
                                 absolute values all array elements.
}

int scan_arr(int* arr){
  int i; char gap;
  for (i = 0, gap = ' '; gap != '\n'; i++)
    scanf("%d%c", &arr[i], &gap);
  return i;
  //Return count of elements.
}
```

```c
void print_err(){
  printf("Данные некорректны\n");
  //Universal error message.
}

int main(){
  //Scan type of task and array of int.
  int type; scanf("%d\n", &type);
  int arr[LIMIT]; int count = scan_arr(arr);

  //Print result of current task.
  switch (type){
    case 0:
      if (index_first_negative(arr, count) < count)
        printf("%d\n", index_first_negative(arr, count));
      else print_err();
      break;
    case 1:
      if (index_last_negative(arr, count) >= 0)
        printf("%d\n", index_last_negative(arr, count));
      else print_err();
      break;
    case 2:
      printf("%d\n", sum_between_negative(arr, count));
      break;
    case 3:
      printf("%d\n", sum_before_and_after_negative(arr, count));
      break;
    default:
      print_err();
  }

  return 0;
}
```