**“ФЕДЕРАЛЬНОЕ АГЕНТСТВО СВЯЗИ**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ТЕЛЕКОММУНИКАЦИЙ ИМ. ПРОФ. М.А. БОНЧ-БРУЕВИЧА”**

**(СПбГУТ)**

**Факультет инфокоммуникационных Сетей и систем (иксс)**

**кафедра программной инженерии и вычислительной техники (пи и вт)**

Дисциплина: «Машинно-зависимые языки программирования»

Лабораторная работа №4.

Тема: «обработка одномерных массивов»

вариант №17

Выполнил: Студент группы ИКПИ-95

Новиков С.С.

Подпись\_\_\_\_\_\_\_\_\_\_\_\_

Принял: Коробов С.А.

Подпись\_\_\_\_\_\_\_\_\_\_\_\_

«\_\_\_\_\_»\_\_\_\_\_\_\_\_ 2020

2020г

**ЦЕЛЬ РАБОТЫ:**

Задав одномерный массив целочисленных данных **А** в одном из заданных форматов (unsigned char — **BYTE**, unsigned short int — **WORD**, char — **SHORTEST**, short int — **INTEGER** или long int — **LONGINT**), реализовать обработку массива, как указано в варианте. Длина массива **N**. Исходные данные задать самостоятельно, учитывая формат элементов массива **А**.  
  
В программе на **C** должны быть предусмотрены функции ввода - вывода элементов массива **А** и его обработки. Исходные данные должны вводиться корректно и быть приближенными к максимальновозможным для данного типа данных. Тип результата определяется из контекста задачи.

**Найти сколько элементов массива A={a[i]} удовлетворяет условию с <= a[i] <= d**

**Таблица Результатов**

|  |  |  |
| --- | --- | --- |
| **c** | **d** | **результат** |
| **-100** | **100** | **10** |
| **3** | **6** | **1** |
| **1** | **22** | **2** |
| **-3** | **3** | **6** |

**Код:**

**Файл main.c:**

#include <stdio.h>

#include <stdint.h>

#include <stdlib.h>

#define len 10

int arr\_size = len;

int64\_t c, d;

\_\_int64\_t my\_array[len];

extern uint64\_t asmF(int64\_t c, int64\_t d);

void cMode() {

uint64\_t counter = 0;

for (uint64\_t i = 0; i < arr\_size; i++) {

if (c <= my\_array[i] && my\_array[i] <= d) {

counter++;

}

}

printf("Проверка из си: %ld\n", counter);

}

void asmMode() {

uint64\_t sum = asmF(c, d);

printf("Результат из nasm: %ld\n", sum);

}

int main() {

for (int64\_t i = 0; i < arr\_size; i++) {

my\_array[i] = (int64\_t) rand() % 21 - 10;

printf("%ld\n", my\_array[i]);

}

printf("Enter c: ");

scanf("%ld", &c);

printf("Enter d: ");

scanf("%ld", &d);

cMode();

asmMode();

return 0;

}

**Файл ASM (NASM):**

SECTION .DATA

extern my\_array

extern my\_array

extern arr\_size

global asmF

asmF:

; 1 - RDI (для windows rcx)

; 2 - RSI (для windows rdx)

; mov r13, rdx ;перенос параметра d (2'ой) в r13

mov r13, rsi ;перенос параметра d (2'ой) в r13

xor r10, r10 ;counter = 0

xor r8, r8 ;i = 0

mov r12, [arr\_size] ;получнеие размера массива

@L1whileIf: ;метка начала цикла

cmp r8, r12 ;i == arr\_size -> выход из цикла

je @L2End

;if

mov rax, 8 ;размер переменной

xor rsi, rsi ;зануление перед \*

mul r8 ;8\*i

mov rbx, my\_array ;получение адреса массива

add rax, rbx ;сложение адреса массива со смещением элемента

mov rbx, [rax] ;получение значения my\_array[i]

cmp rdi, rbx ;c <= a[i]

jg @L3EndIf

cmp rbx, r13 ;a[i] <= d

jg @L3EndIf

inc r10 ;l++

@L3EndIf:

inc r8 ; --i

jmp @L1whileIf ;прыжок в начало цикла

@L2End: ;метка конца цикла

mov rax, r10 ;возврат результата функции

ret