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

# ОТЧЕТ

**по лабораторной работе №6**

**по дисциплине «ОргЭВМис»**

**Тема: Организация связи Ассемблера с ЯВУ на примере программы построения частотного распределение попаданий псевдослучайных целых чисел в заданные интервалы.**

Студент гр. 0382 Гудов Н.Р.

Преподаватель Ефремов М.А.

Санкт-Петербург

2021

**Цель работы.**

Изучение связи Ассемблера с ЯВУ. Написание программы для построения частотного распределения попаданий чисел в интервалы.

**Задание.**

Вар 21.

На языке С программируется ввод с клавиатуры и контроль исходных данных, а также генерируется массив псевдослучайных целых чисел, изменяющихся в заданном диапазоне и имеющих заданный закон распределения. Необходимые датчики псевдослучайных чисел находятся в каталоге RAND_GEN (при его отсутствии получить у преподавателя).

Следует привести числа к целому виду с учетом диапазона изменения.

Далее должны вызываться 1 или 2 ассемблерных процедуры для формирования распределения количества попаданий псевдослучайных целых чисел в заданные интервалы. Ассемблерные процедуры должны вызываться как независимо скомпилированные модули. Передача параметров в процедуру должна выполняться через кадр стека.

Результирующий массив частотного распределения чисел по интервалам, сформированный на ассемблерном уровне, возвращается в программу, реализованную на ЯВУ, и затем сохраняется в файле и выводится на экран средствами ЯВУ.

**Выполнение работы.**

Считывание необходимых данных. Проверка на левую границу и количество интервалов выдает сообщение если они заданы неправильно. Далее происходит генерация чисел согласно равномерному распределению.

Происходит вызов ассемблерной функции, где происходит подсчет вхождений в интервалы. Один модуль состоит из одной функции, принимающей массив и его характеристики. Элемент массива сопоставляется интервалу, после чего счетчик для этого интервала увеличивается.

После возврата из процедуры происходит запись результатов в файл и вывод на экран.

**Тестирование.**

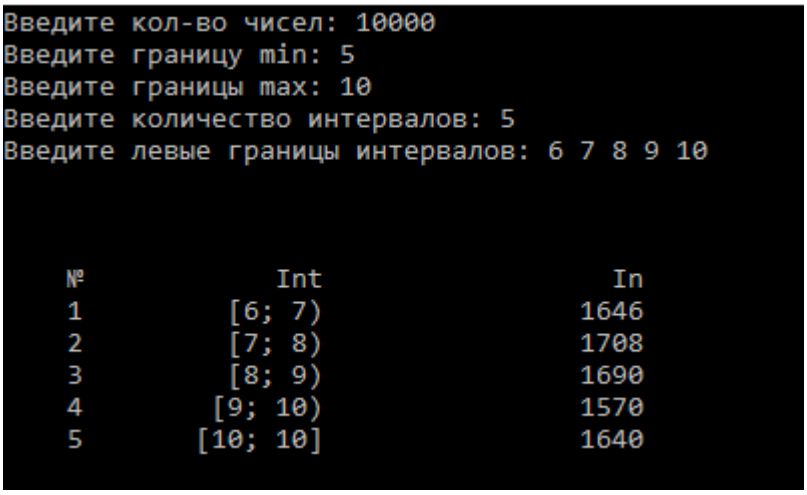


Рис 1. Правильный ввод данных.

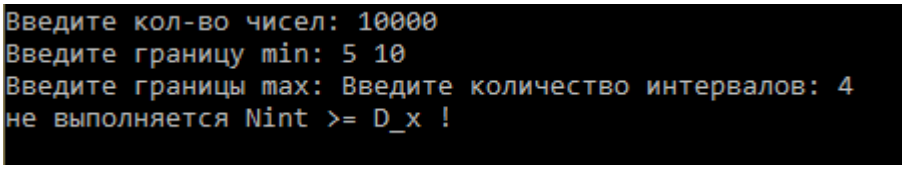


Рис 2. Неправильный ввод данных.

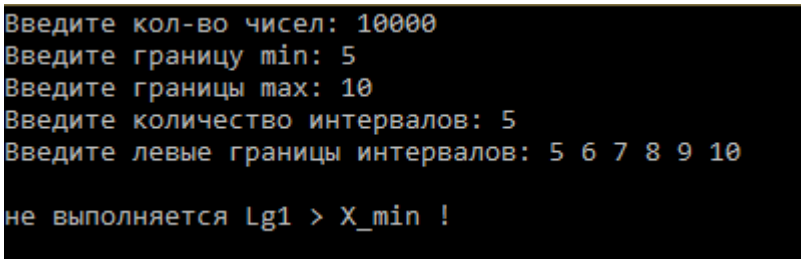


Рис 3. Неправильный ввод данных.

**Выводы.**

Была изучена связь Ассемблера с ЯВУ. Написана программа построения частотного распределения попаданий псевдослучайных чисел в интервалы.

# ПРИЛОЖЕНИЕ А
# ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: Clab6.cpp

```
#include <iostream>
#include <iomanip>
#include <string>
#include <fstream>
#include <random>
#include <stdlib.h>

using namespace std;

extern "C" void func(int* nums, int numsCount, int* leftBorders, int* result);

//21 Равн распр. Nint  ?  Dx. Lg1 > Xmin

int cmp(const void* a, const void* b) {
    return (*(int*)a - *(int*)b);
}

void OUT(string A, string B, string C, ofstream& file) {
    cout << setw(5) << right << A << setw(15) << right << B << setw(20) << right << C << endl;
    file << setw(5) << right << A << setw(15) << right << B << setw(20) << right << C << endl;
}

int main() {
    setlocale(LC_ALL, "");
    int i, randNumCount, max, min, intervalCount;

    cout << "Введите кол-во чисел: ";
    cin >> randNumCount;
    cout << "Введите границу min: ";
    cin >> min;
    cout << "Введите границы max: ";
    cin >> max;
    cout << "Введите количество интервалов: ";
    cin >> intervalCount;
```

```
if (intervalCount < (max - min)) {
    cout << "не выполняется Nint >= D_x !\n";
    return 0;
}

int* leftBorders = new int[intervalCount + 1];
int* result = new int[intervalCount + 1];
leftBorders[intervalCount] = max + 1;
result[intervalCount] = 0;

cout << "Введите левые границы интервалов: ";

for (i = 0; i < intervalCount; i++) {
    cin >> leftBorders[i];
    result[i] = 0;
}
cout << endl;

if (leftBorders[0] <= min) {
    cout << "не выполняется Lg1 > X_min !\n";
    return 0;
}

qsort(leftBorders, intervalCount, sizeof(int), cmp);

random_device rd{};
mt19937 gen(rd());
std::uniform_int_distribution<int> dist(min, max);
int* nums = new int[randNumCount];
for (i = 0; i < randNumCount; i++) {
    nums[i] = round(dist(gen));
}

if (randNumCount <= 100) {
    for (i = 0; i < randNumCount; i++) {
        cout << nums[i] << " ";
    }
}
cout << endl << endl;

func(nums, randNumCount, leftBorders, result);

ofstream file("output.txt");
```

```
        OUT("№", "Int", "In", file);
        for (i = 0; i < intervalCount; i++) {
            if (i == intervalCount - 1) {
                OUT(
                    to_string(i + 1),
                    '['    +    to_string(leftBorders[i])    +    ";    "    +
to_string(leftBorders[i + 1] - 1) + "]",
                    to_string(result[i + 1]),
                    file);
            }
            else {
                OUT(
                    to_string(i + 1),
                    '['    +    to_string(leftBorders[i])    +    ";    "    +
to_string(leftBorders[i + 1]) + ")",
                    to_string(result[i + 1]),
                    file);
            }
        }
        file.close();

        return 0;
    }
```

Название файла: Alab6.asm

```
.586
.MODEL FLAT, C
.CODE

	func	PROC	C	nums:dword,	numsCount:dword,	leftBorders:dword,
result:dword

		push eax
		push ebx
		push ecx
		push edx
		push esi
		push edi

		mov ecx, numsCount
		mov esi, nums
		mov edi, leftBorders

		mov edx, 0
		l:
			mov ebx, [esi+4*edx]
			cmp ebx, [edi]
			jl continue

			mov eax, 0

			searchInterval:

				cmp ebx, [edi+4*eax]
				jl endSearch
				inc eax
				jmp searchInterval

			endSearch:

			mov edi, result
			mov ebx, [edi+4*eax]
			inc ebx
			mov [edi+4*eax], ebx
```

```
                mov edi, leftBorders

                continue:

                inc edx
                loop l

            pop edi
            pop esi
            pop edx
            pop ecx
            pop ebx
            pop eax

            ret
        func ENDP

END
```