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

ОТЧЕТ

по лабораторной работе №6
по дисциплине «Организация ЭВМ и системы»

Тема: Организация связи Ассемблера с ЯВУ на примере программы построения частного распределения попаданий псевдослучайных чисел в званные интервалы.

Студента гр. 1383 _____ Панов М.Ю.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург
2022

Цель работы.

Рассмотреть способ организации связи Ассемблера с ЯВУ. Разработать программу построения частотного распределения попаданий псевдослучайных
целых чисел в заданные интервалы.

Задание.

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.

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.

В cpp файле считывается исходные данные и проверяются на корректность ввода. Генерируется массив псевдослучайных чисел num с помощью srand, массив answer создается в соответствии с количеством интервалов и заполняется счетчиками для каждого из них. В ассемблерный модуль передаются массив псевдослучайных чисел num, массив answer, массив содержащий в себе границы a так же переменные в которых хранятся

количество чисел и количество границ. В ассемблерном модуле берется смещение до элемента в массиве num, после чего взятое значение сравнивается со всеми элементами массива границ. Если число попадает в интервал, увеличивается соответствующий счетчик в answer, после чего выполняется все выше описанное для следующего элемента массива num.

Выводы.

Была реализована связь Ассемблера с ЯВУ а так же разработана программа, осуществляющая построение

частотного распределения попаданий псевдослучайных чисел с равномерным

распределением в заданные интервалы

Исходный код

```
#include <iostream>
#include <fstream>
#include <ctime>
#include <string>

using namespace std;

extern "C" void f(int* inter, int* num, int* answer, int N_int, int N);

int main() {

srand(static_cast<unsigned int>(time(NULL)));

int N, N_int, X_min, X_max;

cout << "amount of numbers:\n";
```

```
cin >> N;

if (N <= 0) {
	cout << "Incorrect amount of numbers!\n";
	return 0;
}

cout << "min value of numbers:\n";
cin >> X_min;

cout << "max value of numbers:\n";
cin >> X_max;

if (X_min >= X_max) {
	cout << "Incorrect X_min and X_max values!\n";
	return 0;
}

cout << "amount of intervals:\n";
cin >> N_int;

if (N_int <= 0 || N_int > 24 ) {
	cout << "Incorrect amount of intervals!\n";
	return 0;
}

cout << "left borders:" << endl;

auto inter = new int[N_int + 1];
```

```
for (int i = 0; i < N_int; i++) {
        cin >> inter[i];

        if (inter[i] < X_min || inter[i] > X_max) {
        cout << "The border should be in the [X_min, X_max] interval!\n";
        return 0;
        }
}



auto num = new int[N];

int rand_val = X_max - X_min + 1;
for (int i = 0; i < N; i++) {
        num[i] = X_min + rand() % rand_val;
}

cout << "The pseudo-random array is:\n";

for (int j = 0; j < N; j++) {
        cout << num[j] << ' ';
}

cout << endl;

auto answer = new int[N_int];

for (int i = 0; i < N_int; i++) {
        answer[i] = 0;
```

```
	}
	f(inter, num, answer, N_int, N);

	ofstream file("out.txt");
	auto str = "N\tBorders\tNumbers amount";
	file << str << endl;
	cout << str << endl;
	for (int i = 0; i < N_int; i++) {
		auto str_res = to_string(i + 1) + "\t" + to_string(inter[i]) + "\t\t" +
to_string(answer[i]) + "\n";
		file << str_res;
		cout << str_res;
	}

	return 0;
	}
```

module:

```
.MODEL FLAT, C
.CODE

f PROC C inter: dword, num: dword, answer: dword, N_int: dword, N: dword

	push eax
   push ebx
   push ecx
   push edi
   push esi
```

```
    mov eax, 0
        mov esi, num


c_loop:
    mov ebx, 0
    iter:
        cmp ebx, N_int
        jge out_cur_iter
        mov ecx, [esi + 4*eax]
        mov edi, inter
        cmp ecx, [edi + 4*ebx]
        jl out_cur_iter
        inc ebx
        jmp iter

    out_cur_iter:
        dec ebx
        mov edi, answer
        mov ecx, [edi + 4*ebx]
        inc ecx
        mov [edi + 4*ebx], ecx

    next_number:
        inc eax
        cmp eax, N
        jg exit

jmp c_loop
```

```
exit:
        pop edx
    pop ecx
    pop ebx
    pop eax
    pop edi
    pop esi
ret
f ENDP
END
```