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 систем»

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

Вариант 1

Студент гр. 1383 _____ Богданов Е.М.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург

2022

Цели работы.

Изучить организацию связи Ассемблера с ЯВУ на примере программы построения частотного распределения попаданий псевдослучайных чисел в заданные интервалы

Задание.

На языке высокого уровня (Pascal или C) генерируется массив псевдослучайных целых чисел, изменяющихся в заданном диапазоне и имеющих равномерное распределение. Необходимые датчики псевдослучайных чисел находятся в каталоге Tasks\RAND_GEN (при его отсутствии программу датчика получить у преподавателя). Далее должен вызываться ассемблерный модуль(модули) для формирования распределения количества попаданий псевдослучайных целых чисел в заданные интервалы. В общем случае интервалы разбиения диапазона изменения псевдослучайных чисел могут иметь различную длину. Результирующий массив частотного распределения чисел по интервалам, формированный на ассемблерном уровне, возвращается в программу, реализованную на ЯВУ, и затем сохраняется в файле и выводится на экран средствами ЯВУ. Исходные данные:

1. Длина массива псевдослучайных целыхчисел - NumRanDat (<= 16K, K=1024)

2. Диапазон изменения массива псевдослучайных целых чисел [Xmin, Xmax] , значения могут быть биполярные;

3. Количество интервалов, на которые разбивается диапазон изменения массива псевдослучайных целых чисел - NInt (<=24)

4. Массив левых границ интервалов разбиения LGrInt (должны принадлежать интервалу [Xmin, Xmax]).

Результаты:

1. Текстовый файл, строка которого содержит: - номер интервала, - левую границу интервала, - количество псевдослучайных чисел, попавших в интервал. Количество строк равно числу интервалов разбиения.

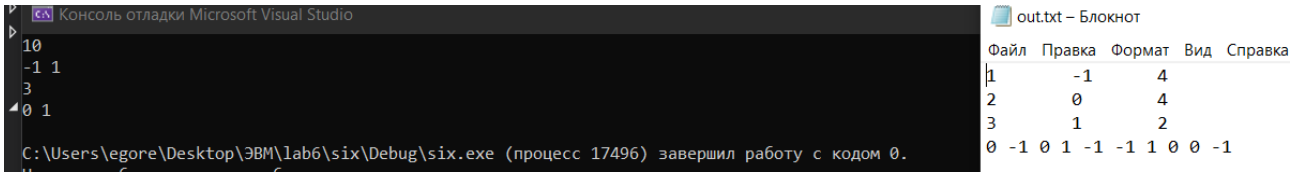
2. График, отражающий распределение чисел по интервалам. (необязательный результат)

Подпрограмма формирования распределения количества попаданий псевдослучайных целых чисел в заданные интервалы реализуется в виде одного ассемблерного модуля, сразу формирующего требуемое распределение и возвращающего его в головную программу, написанную на ЯВУ.

Выполнение работы.

На языке C++ был организован ввод входных данных и генерация псевдослучайного массива. Затем вызывается ассемблерный модуль counter, считающий сколько чисел попал в каждый из интервалов. Далее в файл выводится результат: порядковые номера интервалов, их левые границы, кол-во чисел, попавших в них и псевдорандомный массив.

Рисунок 1 — Пример работы программы



Программный код см. в приложении А.

Выводы.

В ходе лабораторной работы была изучена связь ассемблера с ЯВУ.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

L6.cpp

```
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>
#include <stdlib.h>
#include <iostream>
#include <time.h>
#include <math.h>

using namespace std;

extern "C" void counter(int* array, int array_size, int* LGrInt, int NInt, int*
result_array);

int main()
{
	srand(time(0));
	int NumRanDat, Xmin, Xmax, NInt;
	cin >> NumRanDat >> Xmin >> Xmax >> NInt;
	int* arr = new int[NumRanDat];
	for (int i = 0; i < NumRanDat; ++i)
	{
		arr[i] = Xmin + rand() % abs(Xmax - Xmin+1);
	}

	int* LGrInt = new int[NInt];
	LGrInt[0] = Xmin;
	for (int i = 1; i < NInt; ++i)
	{
		cin >> LGrInt[i];
	}
	int* ans = new int[NInt] {0};
	counter(arr, NumRanDat, LGrInt, NInt, ans);
	freopen("out.txt", "w", stdout);
	for (int i = 0; i < NInt; ++i)
	{
		cout << i + 1 << "   " << LGrInt[i] << "  " << ans[i] << endl;;
	}
	for (int i = 0; i < NumRanDat; ++i)
	{
		cout << arr[i] << " ";
	}
	return 0;
}
```

Lab6.asm

```
.686p
.MODEL FLAT, C
.CODE
counter PROC C USES EDI ESI, arr:dword, siz:dword, LGrInt:dword, NInt:dword,
ans:dword
	push eax
	push ebx
	push ecx
	push edi
	push esi
```

```
	mov ecx, siz
	mov esi, arr
	mov edi, LGrInt
	mov eax, 0

	CYCLE:
		mov ebx, 0
	SEARCH:
		cmp ebx, NInt
		jge O

		push eax
		mov eax, [esi + 4 * eax]

		cmp eax, [edi + 4 * ebx]
		pop eax
		jl O

		add ebx,1
		jmp SEARCH

	O:
		sub ebx,1
		cmp ebx, -1
		je NUMBER

		mov edi, ans
		push eax

		mov eax, [edi + 4 * ebx]
		add eax,1

		mov [edi + 4 * ebx], eax

		pop eax
		mov edi, LGrInt

	NUMBER:
		add eax,1

	loop CYCLE

	pop esi
	pop edi
	pop ecx
	pop ebx
	pop eax

ret

counter ENDP
	END
```