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.

Студентка гр. 9383 ____ Орлов Д.С.

Преподаватель ____ Ефремов М.А.

Санкт-Петербург
2020

Цель работы.

Научиться реализовывать связь Ассемблера и ЯВУ. Написать программу 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, сформированный 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]).

Для бригад с четным номером: подпрограмма формирования распределения количества попаданий псевдослучайных целых чисел в заданные интервалы реализуется в виде двух ассемблерных модулей, первый из которых формирует распределение исходных чисел по интервалам единичной длины и возвращает его в вызывающую программу на ЯВУ как промежуточный результат. Это распределение должно выводиться в текстовом виде для контроля. Затем вызывается второй ассемблерный модуль, который по этому промежуточному распределению формирует окончательное распределение псевдослучайных целых чисел по интервалам произвольной длины (с заданными границами).

Ход работы.

В ходе работы была реализована программа из 3-х модулей, 1 на C++ (ЯВУ) и 2 других на ассемблере.

На ЯВУ написан main.cpp, который собирает от пользователя входную информацию и перенаправляет ее в ассемблерные модули. Так же здесь осуществляется вывод данных в консоль и файл.

На ассемблере написано 2 модуля. Первый реализует распределение чисел по единичным отрезкам. Это сделано с помощью команды loop. Циклически записывается в новый массив количество повторений каждого числа.

Второй модуль формирует распределение тех же чисел, но уже по заданным интервалам. Это происходит благодаря нескольким циклам, в которых левые границы переводятся в неотрицательные числа и сопоставляются числам с таким же индексом из массива, полученного в первом модуле.

Связь между модулями осуществлена с помощью спецификатора extern, который позволяет выполнять раздельную компиляцию модулей.

Исходный код см. в приложении А.

Тестирование.

№	Исходные данные	Результат
1	NumDatRan=100 xmin=-50 xmax=50 NInt=10 LGrInt={-50, -43, -30, 0, 10, 12, 15, 30, 40, 49 }	№ Лев.гр. Кол-во чисел 1 -50 4 2 -43 14 3 -30 28 4 0 7 5 10 1 6 12 6 7 15 18 8 30 11 9 40 11 10 49 0
2	lenArr = 40 xmin= -20 xmax= 20 NInt= 4 LGrInt={-20, -10, 0, 10 }	№ Лев.гр. Кол-во чисел 1 -20 15 2 -10 7 3 0 11 4 10 7
3	lenArr =1100 xmin=0 xmax=100 NInt=7 LGrInt={0, 5, 23, 56, 70, 75, 90}	№ Лев.гр. Кол-во чисел 1 0 48 2 5 211 3 23 360 4 56 149 5 70 56 6 75 155 7 90 121
4	lenArr =80 xmin=0	№ Лев.гр. Кол-во чисел 1 0 14

	xmax=10 NInt=5 LGrInt={ 0, 2, 4, 6, 8}	2 2 13 3 4 13 4 6 15 5 8 25
5	lenArr =16000 xmin=-8 xmax=8 NInt=6 LGrInt={ -8 -4 -2 0 3 6}	№ Лев.гр. Кол-во чисел 1 -8 3956 2 -4 1982 3 -2 2071 4 0 2983 5 3 3002 6 6 2006

Выводы.

Была реализована связь модуля на ЯВУ и модулей на Ассемблере.

ПРИЛОЖЕНИЕ А.

ИСХОДНЫЙ КОД ПРОГРАММЫ.

Название файла: lab6.cpp

```
#include <iostream>
#include <fstream>
#include <random>

using namespace std;

extern "C" void first(int* numbers, int numbers_size, int* result, int xmin);
extern "C" void second(int* array, int array_size, int xmin, int* intervals, int
intervals_size, int* result);


int main()
{
	setlocale(0, "Russian");
	srand(time(NULL));
	ofstream result("result.txt");

	int numbers_size;
	int* numbers;
	int xmin, xmax;
	int intervals_size;
	int* intervals;
	int* intervals2;
	int* mod1_result;
	int* mod2_result;

	cout << "Введите количество чисел:\n";
	cin >> numbers_size;
	if (numbers_size > 16 * 1024)
	{
		cout << "Количество чисел должно быть меньше или равно, чем 16*1024\n";
		return 0;
	}
	cout << "Введите xmin и xmax:\n";
	cin >> xmin >> xmax;
	cout << "Введите число границ:\n";
	cin >> intervals_size;
	if (intervals_size > 24)
	{
		cout << "Число интервалов должно быть меньше или равно 24\n";
		return 0;
	}

	numbers = new int[numbers_size];
	intervals = new int[intervals_size];
	intervals2 = new int[intervals_size];

	int len_asm_mod1_res = abs(xmax - xmin) + 1;
	mod1_result = new int[len_asm_mod1_res];
	for (int i = 0; i < len_asm_mod1_res; i++)
	{
		mod1_result[i] = 0;
	}

	mod2_result = new int[intervals_size+1];
	for (int i = 0; i < intervals_size+1; i++)
	{
		mod2_result[i] = 0;
	}

	cout << "Введите все границы:\n";
	for (int i = 0; i < intervals_size; i++)
```

```
	{
		cin >> intervals[i];
		intervals2[i] = intervals[i];
	}



	for (int i = 0; i < numbers_size; i++)
	{
		numbers[i] = xmin + rand() % (xmax - xmin + 1) ;
	}

	first(numbers, numbers_size, mod1_result, xmin);

	second(mod1_result, numbers_size, xmin, intervals, intervals_size,
mod2_result);


	cout << "Результат:\n";
	result << "Результат:\n";
	cout << "№\tГраница\tКоличество чисел" << endl;
	result << "№\tГраница\tКоличество чисел" << endl;

	for (int i = 1; i < intervals_size + 1; i++)
	{
		if (i != intervals_size)
		{
			cout << i  << "\t" << intervals2[i-1] << '\t' << mod2_result[i]
<< endl;
			result << i << "\t" << intervals2[i-1] << '\t' << mod2_result[i]
<< endl;
		}
		else
		{
			cout << i << "\t" << xmax << '\t' << mod2_result[i] << endl;
			result << i << "\t" << xmax << '\t' << mod2_result[i] << endl;
		}
	}

	delete[] numbers;
	delete[] intervals;
	delete[] intervals2;
	delete[] mod1_result;
	delete[] mod2_result;

	return 0;
}
```

Название файла: first.asm

```
.586p
.MODEL FLAT, C
.CODE
PUBLIC C first
first PROC C array: dword, arraysize: dword, res: dword, xmin: dword

push esi
push edi

mov edi, array  ;исходный массив
mov ecx, arraysize
mov esi, res  ;массив на выход

for_numbers:
	mov eax, [edi]
	sub eax, xmin
	mov ebx, [esi + 4*eax]
	inc ebx
	mov [esi + 4*eax], ebx
	add edi, 4
```

```
      loop for_numbers

pop edi
pop esi

ret
first ENDP
EN
```

Название файла: second.asm

```
.586p

.MODEL FLAT, C

.CODE

PUBLIC C second

second PROC C array: dword, array_size: dword, xmin: dword, borders: dword,
intN: dword, result: dword


push esi

push edi

push ebp


mov edi, array

mov esi, borders

mov ecx, intN



for_borders:

   mov eax, [esi]

   sub eax, xmin

   mov [esi], eax

   add esi, 4

   loop for_borders



mov esi, borders

mov ecx, intN

mov ebx, 0

mov eax, [esi]


for_loop:
```

```
   push ecx
   mov ecx, eax
   push esi
   mov esi, result

    for_array:
         cmp ecx, 0
         je end_for
        mov eax, [edi]
        add [esi + 4*ebx], eax
        add edi, 4
        loop for_array

end_for:
    pop esi
    inc ebx
   mov eax, [esi]
   add esi, 4
   sub eax, [esi]
   neg eax
   pop ecx
   loop for_loop

mov esi, result
mov ecx, intN
mov eax, 0

fin_for:
   add eax, [esi]
   add esi, 4
   loop fin_for

mov esi, result
sub eax, array_size
neg eax
```

```
add [esi + 4*ebx], eax

pop ebp
pop edi
pop esi

ret
second ENDP
END
```