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.

Студент гр. 9382 _____ Кузьмин Д. И.

Преподаватель _____ Ефремов М. А.

Санкт-Петербург
2020

Цель работы.

Изучить принципы работы со массивами в ассемблере. Освоить навыки разработки программ на ЯВУ со включением кода на Ассемблере.

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.

Вариант 2: 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). Это распределение возвращается в головную программу и выдается как основной результат в виде текстового файла и, возможно, графика.

Выполнение работы.

1) Первым шагом было объявление различных переменных, а также реализация ввода данных.

2) Был создан массив случайных чисел, массив левых границ интервалов, массив частот единичных интервалов, массив частот заданных интервалов.

3) Затем при помощи вставки ассемблерного кода был реализован расчет частоты попадания сгенерированных случайных чисел в единичные интервалы.

4) Далее следовал вывод этих частот.

5) На следующем шаге был реализован на ассемблере расчет частот попадания чисел в заданные интервалы и их вывод.

Тестирование.

Результаты тестирование представлены в табл. 1.

Таблица 1 — результаты тестирования.

№ п/п	Входные данные	Выходные данные	Комментарий
1	размер массива: 5 диапазон распределения: -2 2 количество интервалов: 3 Левые границы интервалов: -1 0 1	Массив случайных чисел: -1 0 2 -2 2 Распределение по единичным интервалам: [-2]: 1 [-1]: 1 [0]: 1 [1]: 0 [2]: 2 Распределение по заданным интервалам: [-1,-1]: 1 [0,0]: 1 [1,2]: 2	Распределение по 3 интервалам.
2	размер массива: -5	Повторите ввод:	Программа обрабатывает некорректный размер массива

3	размер массива: 5 диапазон распределения: 0 10 количество интервалов: 1 Левые границы интервалов: 0	Массив случайных чисел: 8 9 9 1 7 Распределение по единичным интервалам: [0]: 0 [1]: 1 [2]: 0 [3]: 0 [4]: 0 [5]: 0 [6]: 0 [7]: 1 [8]: 1 [9]: 2 [10]: 0 Распределение по заданным интервалам: [0,10]: 5	Случай, когда интервал всего один.

Выводы.

Были изучены принципы связи Ассемблера с ЯВУ. Получены навыки разработки программ, работающих с массивами чисел.

ПРИЛОЖЕНИЕ А. ИСХОДНЫЙ КОД

Файл main.cpp

```
#include <iostream>

int main()
{

	int n;
	int xmin;
	int xmax;
	int arr_length;

	setlocale(LC_ALL, "Russian");

	std::cout << "Введите размер массива: ";
	std::cin >> n;
	while (n <= 0) {
		std::cout << "Повторите ввод:";
		std::cin >> n;
	}
	int* randomnumbers = new int[n];

	std::cout << "Введите диапазон распределения: ";
	std::cin >> xmin >> xmax;
	while (xmin > xmax) {
		std::cout << "Повторите ввод:";
		std::cin >> xmin >> xmax;
	}
	const int range = abs(xmax - xmin) + 1;
	int* unitintervalfrequency = new int[range];


	std::cout << "Введите количество интервалов: ";
	std::cin >> arr_length;
	while (arr_length <= 0) {
		std::cout << "\nПовторите ввод: ";
		std::cin >> arr_length;
	}
	int* lgrint = new int[arr_length];
	int* frequency = new int[arr_length];


	std::cout << "Левые границы интервалов: ";
	for (int i = 0; i < arr_length; i++){
		std::cin >> lgrint[i];
		frequency[i] = 0;
	}


	std::cout << "Разделение на интервалы: ";
	for (int i = 0; i < arr_length; i++) {
		std::cout << "[" << lgrint[i] << ",";
		if (i != arr_length - 1) std::cout << lgrint[i + 1] - 1 <<
"]";
		else std::cout << xmax << "]";
```

```
	}
	std::cout << "\nМассив случайных чисел: ";
	for (int i = 0; i < n; i++) {
		randomnumbers[i] = rand() % (abs(xmax - xmin) + 1) + xmin;
		std::cout << randomnumbers[i] << " ";

	}
	for (int i = 0; i < range; i++) unitintervalfrequency[i] = 0;
	std::cout << "\nРаспределение по единичным интервалам:\n";
	int h;
	_asm {

	mov ecx, n
	sub ecx, 1
	sub ebx, ebx
	mov edx, randomnumbers
	mov eax, unitintervalfrequency
	unit:
	mov esi, [edx + ebx*4]
	sub esi, xmin
	add [eax + esi*4], 1
	cmp ebx, ecx
	jge endloop
	inc ebx
	jmp unit
	endloop:

	}
	for (int i = 0; i < range; i++)
	std::cout << "[" << i + xmin << "]: " << unitintervalfrequency[i]
<< "\n";
	_asm {

		mov eax, unitintervalfrequency
		mov edx, frequency

		mov ecx, arr_length
		sub edi, edi
		cmp ecx, 1
		je lastint

		sub ecx, 2


		mov esi, lgrint
		mov esi, [esi + edi*4];
		sub esi, xmin

		mov ebx, lgrint
		add edi, 1
		mov ebx, [ebx + edi*4]
		sub ebx, xmin
		sub edi, 1



		countinterval:
		cmp esi, ebx
```

```
            je nextinterval

            mov eax, [eax + esi * 4]
            add [edx + edi*4], eax
            mov eax, unitintervalfrequency

            inc esi

            jmp countinterval

            nextinterval:
            cmp edi, ecx
            je lastint
            jg endprog
            inc edi
            mov esi, lgrint
            mov esi, [esi + edi*4]
            sub esi, xmin
            mov ebx, lgrint
            add edi, 1
            mov ebx, [ebx + edi*4]
            sub edi,  1
            sub ebx, xmin

            jmp countinterval

            lastint:
            cmp ecx, 1
            je noinc
            inc edi
            noinc:
            mov ecx, 0
            mov esi, xmax
            sub esi, xmin
            add esi, 1
            cmp ebx, esi
            je endprog
            mov ebx, xmax
            sub ebx, xmin
            add ebx, 1
            mov esi, lgrint
            mov esi, [esi + edi*4]
            sub esi, xmin
            jmp countinterval

            endprog:
      }
      std::cout << "Распределение по заданным интервалам:\n";
      for (int i = 0; i < arr_length; i++)
            if (i != arr_length - 1)std::cout << "[" << lgrint[i] << ","
<<lgrint[i + 1] - 1 << "]: " << frequency[i] << "\n";
            else std::cout << "[" << lgrint[i] << "," << xmax << "]: " <<
frequency[i] << "\n";
}
```