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

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 14

Студент гр. 1303 _____ Кузнецов Н.А.

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.

Main.cpp

В функции main() происходит считывание данных и их подготовка к передаче в ассемблерный блок. А также обрабатываются следующие необходимые по условию исключения:

1) Количество интервалов должно быть от 0 до 24
2) Количество интервалов должно быть больше или равно Xmax - Xmin
3) Хотя бы одна левая граница должна быть меньше или равна Xmin

4) Правая граница последнего интервала должна быть меньше либо равна Xmax

После вызова ассемблерного блока выводится результат его выполнения.

Module.asm

В блоке происходит поиск количества чисел, вошедших в каждый заданный интервал. Поиск реализован в виде двойного цикла. В первом происходит последовательный выбор из массива сгенерированных чисел. В во втором - перебор левых границ для нахождения нужного диапазона. Перед вторым циклом выбранное число проверяется на вхождение в диапазон между правой границей и Xmax, в случае положительного исхода - число пропускается, так как оно не попадает ни в один из диапазонов. Иначе левые границы перебираются в порядке убывания. После выхода из внутреннего цикла из массива final_answer для вывода достается число, соответствующее найденному диапазону, увеличивается на один и кладется обратно. После прохождения циклов полученный массив final_answer будет использоваться для вывода.

Тестирование.

Тест 1.

```
Введите количество чисел:
20
Введите диапазон генерации:
-5 5
Введите количество интервалов:
11
Введите левые границы:
-10 -9 -4 -8 -7 -6 -2 0 3 2 1
Введите правую границу:
4
0 0 1 3 4 4 2 0 5 2 -3 -3 5 -1 4 3 -3 0 1 -4
N       Borders The Amount of numbers
1       -10             0
2       -9              0
3       -8              0
4       -7              0
5       -6              0
6       -4              4
7       -2              1
8       0               4
9       1               2
10      2               2
11      3               5
```

Тест 2.

```
Введите количество чисел:
10
Введите диапазон генерации:
-10 0
Введите количество интервалов:
3
Количество интервалов должно быть больше |Xmax - Xmin|
Для продолжения нажмите любую клавишу . . .
```

Тест 3.

```
Введите количество чисел:
3
Введите диапазон генерации:
-2 1
Введите количество интервалов:
3
Введите левые границы:
-1 0 1
Хотя бы одна левая граница должна быть меньше чем Xmin
Для продолжения нажмите любую клавишу . . .
```

Вывод.

Рассмотрен способ организации связи Ассемблера с ЯВУ. Составлена программа построения частотного распределения попаданий псевдослучайных целых чисел в заданные интервалы.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.cpp

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

using namespace std;

extern "C" void func(int* intervals, int N_int, int N, int* numbers,
int* final_answer);

int main() {
	system("chcp 1251 > nul");
	setlocale(LC_CTYPE, "rus");

	int N, X_min, X_max, N_int;
	cout << "Введите количество чисел:" << endl;
	cin >> N;
	cout << "Введите диапазон генерации:" << endl;
	cin >> X_min >> X_max;
	cout << "Введите количество интервалов:" << endl;
	cin >> N_int;

	if (N_int <= 0 || N_int > 24) {
		cout << "Количество интервалов должно быть от 0 до 24" <<
endl;
		system("Pause");
		return 0;
	}

	if (N_int < abs(X_max - X_min)) {
		cout << "Количество интервалов должно быть больше или равно
Xmax - Xmin" << endl;
		system("Pause");
		return 0;
	}

	cout << "Введите левые границы:" << endl;
	auto intervals = new int[N_int + 1];
	for (int i = 0; i < N_int; ++i) {
		cin >> intervals[i];
	}

	for (int i = 0; i < N_int; i++) {
		for (int j = i; j < N_int; j++) {
			if (intervals[i] > intervals[j]) {
				swap(intervals[i], intervals[j]);
			}
		}
	}
```

```
	if (intervals[0] > X_min) {
		cout << "Хотя бы одна левая граница должна быть меньше или равна Xmin" << endl;
		system("Pause");
		return 0;
	}

	cout << "Введите правую границу:" << endl;
	cin >> intervals[N_int];

	if (intervals[N_int] < intervals[N_int - 1]) {
		cout << "Правая граница последнего интервала должна быть больше левой границы последего интервала" << endl;
		system("Pause");
		return 0;
	}

	if (intervals[N_int] > X_max) {
		cout << "Правая граница последнего интервала должна быть меньше или равна X_max!" << endl;
		system("Pause");
		return 0;
	}

	auto numbers = new int[N];
	random_device rd;
	mt19937 generator(rd());
	uniform_int_distribution<> dist(X_min, X_max);
	for (int i = 0; i < N; i++) {
		numbers[i] = dist(generator);
		cout << numbers[i] << " ";
	}
	cout << endl;

	auto final_answer = new int[N_int];

	for (int i = 0; i < N_int; i++) {
		final_answer[i] = 0;
	}
	func(intervals, N_int, N, numbers, final_answer);

	ofstream file("output.txt");
	auto str = "N\tBorders\tThe Amount of numbers";
	file << str << endl;
	cout << str << endl;
	for (int i = 0; i < N_int; i++) {
		auto str_res = to_string(i + 1) + "\t" + to_string(intervals[i]) + "\t\t" + to_string(final_answer[i]) + "\n";
		file << str_res;
		cout << str_res;
	}
	system("Pause");
	return 0;
}
```

Название файла: module.asm

```
.MODEL FLAT, C
.CODE

PUBLIC C func
func PROC C intervals: dword, N_int: dword, N: dword, numbers: dword,
final_answer: dword
      push esi
      push edi
      push eax
      push ebx
      push ecx
      push edx

      mov esi, numbers
      mov edi, final_answer
      mov eax,0

      begin:
            mov ebx,[esi + 4*eax] ;берем   текущее   число   из   массива
сгенерированных чисел
            push esi
            mov ecx, N_int

            mov esi, intervals
            cmp [esi + 4*ecx],ebx   ;проверка на то, что число находится
между правой границей и Xmax
            jl ending               ;в случае если это так - пропускаем
его
            dec ecx
            border_begin:
                  cmp ebx, [esi + 4*ecx]  ;проверка на то что, взятое
число больше следующей левой границы
                  jge print_result;в случае если это так -  переходим к
его записи
                  dec ecx
                  jmp border_begin

            print_result:         ;запись числа в массив вывода
                  mov esi,final_answer
                  mov ebx,[esi + 4*ecx]
                  inc ebx
                  mov [edi + 4*ecx], ebx

            ending:
                  pop esi
                  inc eax
                  cmp eax,N               ;проверка   на   то   что,   весь
массив чисел обработан
                  jne begin
```

```
      pop edx
      pop ecx
      pop ebx
      pop eax
      pop edi
      pop esi
ret
func ENDP
END
```