**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.

|  |  |  |
| --- | --- | --- |
| Студент гр. 9382 |  | Рыжих Р.В. |
| Преподаватель |  | Ефремов М.А. |

Санкт-Петербург

2020

**Цель работы.**

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

**Задание.**

На языке высокого уровня (Pascal или С) генерируется массив псевдослучайных целых чисел, изменяющихся в заданном диапазоне и имеющих равномерное распределение. Необходимые датчики псевдослучайных чисел находятся в каталоге Tasks\RAND\_GEN (пpи 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 целых чисел - NumRanDat (<= 16K, К=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)

В зависимости от номера бригады формирование частотного распределения должно производиться по одному из двух вариантов:

1. Для бригад с нечетным номером: 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;

**Ход работы:**

В качестве ЯВУ используется язык С++.

Пользователь вводит необходимые данные в программу: длина массива, нижняя и верхняя границы значений, количество интервалов и нижние границы интервалов. Далее генерируется массив из псевдослучайных целых чисел, который передается в ассемблерный модуль для формирования распределения количества попаданий этих чисел в заданные интервалы.

В завершении на экран выводится результат работы программы; также результат вносится в текстовый файл result.txt.

**Тестирование.**

Рисунок 1 — Пример работы программы с входными данными №1

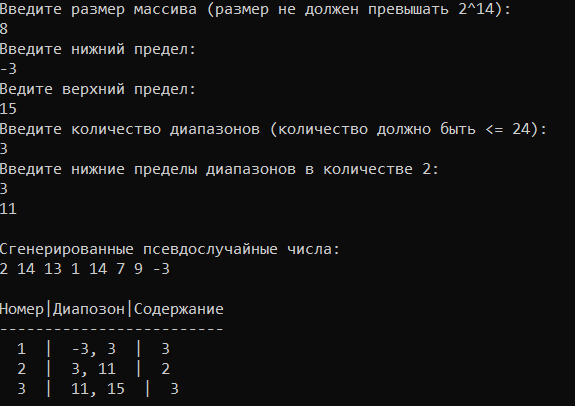


Рисунок 2 — Пример работы программы с входными данными №2

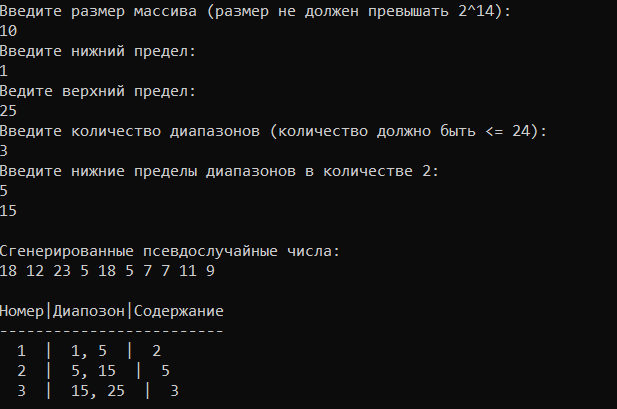
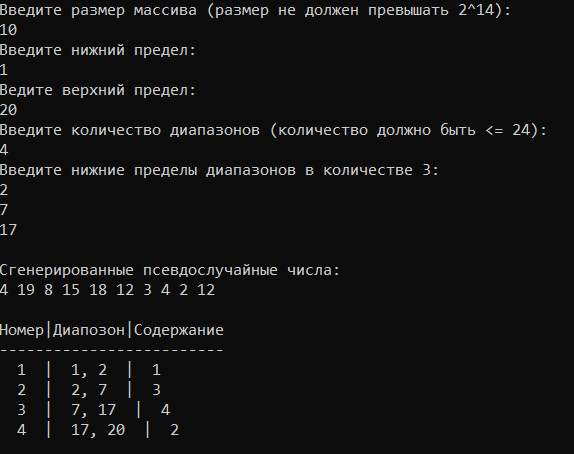


Рисунок 3 — Пример работы программы с входными данными №3



**Выводы.**

В результате выполнения лабораторной работы была изучена организация связи Ассемблера с ЯВУ и написана программа построения частотного распределения попаданий псевдослучайных целых чисел в заданные интервалы.

Исходный код программы см. в приложении А.

**ПРИЛОЖЕНИЕ А**

**ИСХОДНЫЙ КОД ПРОГРАММЫ**

**Файл module.asm:**

.686

.MODEL FLAT, C

.STACK

.DATA

.CODE

distribution PROC C capacity: dword, arr: dword, LGrInt: dword, range: dword

mov ecx, 0 ; счетчик для прохода по массиву

mov ebx, [arr] ; входной массив

mov esi, [LGrInt] ; массив с левыми границами

mov edi, [range] ; результат

f1:

mov edx, [ebx]; берем элемент входного массива

push ebx; сохраняем указатель на текущий элемент

sub ebx, ebx; обнуляем указатель

f2:

mov eax, ebx; eax содержит текущий индекс массива границ

shl eax, 2 ; индекс умножаем на 4, так как каждый элемент по 4 байт

cmp edx, [esi+eax] ; сравниваем текующий элемент с текущей левой границей

jle fe

inc ebx

jmp f2

fe:

add eax, edi ; после сложения указываем на элемент в результирующем массиве для инкрементирования

mov edx, [eax]

inc edx

mov [eax], edx

pop ebx ; забираем текущий элемент и ссылаемся на новый

add ebx, 4

inc ecx ; инкрементируем индекс массива

cmp ecx, capacity

jl f1

ret

distribution ENDP

END **Файл lab6.cpp:**

#include <iostream>

#include <stdlib.h>

#include <fstream>

using namespace std;

extern "C" {

void distribution(int capacity, int\* arr, int\* LGrInt, int\* range);

}

int main() {

setlocale(LC\_ALL, "Russian");

int capacity = 0, Xmin = 0, Xmax = 0, NInt = 0;

cout << "Введите размер массива (размер не должен превышать 2^14):\n";

cin >> capacity;

if (capacity > 16 \* 1024 || capacity <= 0) {

cout << "Ошибка! Неверный размер массива.\n";

exit(1);

}

cout << "Введите нижний предел:\n";

cin >> Xmin;

cout << "Ведите верхний предел:\n";

cin >> Xmax;

if (Xmin > Xmax) {

cout << "Верхний и нижний пределы поменяны местами!\n";

swap(Xmin, Xmax);

}

cout << "Введите количество диапазонов (количество должно быть <= 24): \n";

cin >> NInt;

if (NInt > 24 || NInt < 1 || NInt > (Xmax - Xmin + 1)) {

cout << "Ошибка! Неверное количество диапазонов.\n";

exit(1);

}

int\* LGrInt = new int[NInt]();

cout << "Введите нижние пределы диапазонов в количестве " << NInt - 1 << ":\n";

for (int i = 0; i < NInt - 1; i++) {

cin >> LGrInt[i];

if (LGrInt[i] < LGrInt[i - 1]) {

cout << "Введенный предел " << LGrInt[i] << " больше предыдущего\n";

cin >> LGrInt[i];

}

if (LGrInt[i] < Xmin || LGrInt[i] > Xmax) {

cout << "Ошибка! Неверный нижний предел.\n";

exit(1);

}

}

LGrInt[NInt - 1] = Xmax;

int\* arr = new int[capacity]();

for (int i = 0; i < capacity; i++) {

arr[i] = Xmin + rand() % (Xmax - Xmin);

}

int\* range = new int[NInt];

for (int i = 0; i < NInt; i++)

range[i] = 0;

distribution(capacity, arr, LGrInt, range);

ofstream file("result.txt");

if (!file.is\_open())

cout << "Невозможно открыть файл!\n";

cout << "\nСгенерированные псевдослучайные числа:\n";

file << "Сгенерированные псевдослучайные числа:\n";

for (int i = 0; i < capacity; i++) {

cout << arr[i] << " ";

file << arr[i] << " ";

}

cout << "\n\n";

file << "\n\n";

cout << "Номер|Диапазон|Содержание\n";

file << "Номер|Диапазон|Содержание\n";

cout << "-------------------------" << endl;

file << "-------------------------" << endl;

int n1, n2;

for (int i = 0; i < NInt; i++) {

if (i == 0) {

n1 = Xmin;

n2 = LGrInt[i];

}

else

{

n1 = LGrInt[i - 1];

n2 = LGrInt[i];

}

file << " " << i + 1 << " | " << n1 << ", " << n2 << " | " << range[i] << "\n";

cout << " " << i + 1 << " | " << n1 << ", " << n2 << " | " << range[i] << "\n";

}

file.close();

}