**МИНОБРНАУКИ РОССИИ**

**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**

**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**

**Кафедра МОЭВМ**

**ОТЧЕТ**

**по лабораторной работе №5**

**по дисциплине «Организация ЭВМ и систем»**

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

| Студент гр. 1303 |  | Чубан Д.В. |
| --- | --- | --- |
| Преподаватель |  | Ефремов М.А. |

Санкт-Петербург

2022

**Цель работы.**

Рассмотреть способ организации связи ассемблера с ЯВУ на примере связи с языком программирования С++. Разработать программу, выполняющую подсчет попаданий псевдослучайных 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. Необходимые датчики псевдослучайных чисел находятся в каталоге RAND\_GEN (пpи его отсутствии получить у пpеподавателя). Следует привести числа к целому виду с учетом диапазона изменения. 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.**

В ходе выполнения работы, написаны два исходных файла – основное тело программы, описанное в файле main.cpp, и модуль, выполняющий обработку данных, написанный на языке ассемблера.

В функции main() происходит считывание входных данных, а так же проверка ввода на корректность. Так же в данной функции происходит подготовка 3 данных для передачи их в ассемблерный модуль. С помощью функции стандартной библиотеки std::uniform\_int\_distribution<int> генерируется массив псевдослучайных чисел с нормальным распределением Гаусса. Затем все сгенерированные числа передаются в ассемблерный блок. Полученные результаты выводятся на экран и в файл средствами ЯВУ.

В ассемблерном модуле обработка осуществляется следующим образом:

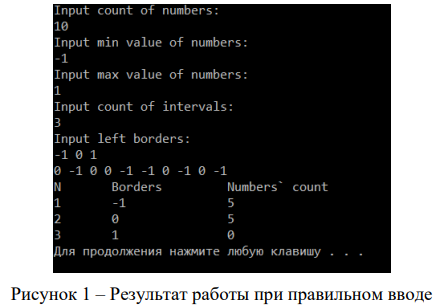
1. В регистр ECX помещается смещение до очередного числа из сгенерированного массива

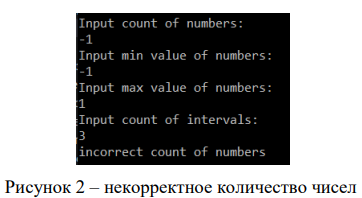
2. Циклом перебираем левые границы интервалов и сравниваем его с числом, помещенным в ECX.

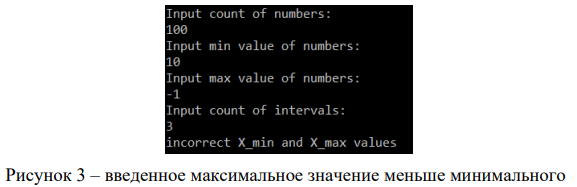
3. В случае, если число попадает в какой-то из интервалов – увеличивается значение в массиве final\_answer[], соответствующее данному интервалу.

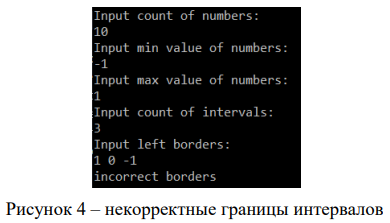
4. В ином случае, просто переходим к следующему числу.

**Тестирование.**

****

****

****

****

**Вывод.**

Рассмотрен способ организации связи ассемблера с ЯВУ. Разработана программа, строящая частотное распределение попадания псевдослучайных чисел, сгенерированных с нормальным распределением, 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() {

int N, X\_min, X\_max, N\_int;

cout << "Input count of numbers:\n";

cin >> N;

cout << "Input min value of numbers:\n";

cin >> X\_min;

cout << "Input max value of numbers:\n";

cin >> X\_max;

cout << "Input count of intervals:\n";

cin >> N\_int;

if (N <= 0) {

cout << "incorrect count of numbers\n";

return 0;

}

if (X\_min >= X\_max) {

cout << "incorrect X\_min and X\_max values\n";

return 0;

}

if (N\_int <= 0 || N\_int > 24) {

cout << "incorrect count of intervals\n";

return 0;

}

cout << "Input left borders:" << endl;

auto intervals = new int[N\_int + 1];

for (int i = 0; i < N\_int; ++i) {

cin >> intervals[i];

if (intervals[i] < X\_min || intervals[i] > X\_max) {

cout << "border should be in the [X\_min, X\_max] interval\n";

delete[]intervals;

return 0;

}

}

cout << "input right border:\n";

cin >> intervals[N\_int];

for (int i = 0; i < N\_int - 1; i++) {

for (int j = i + 1; j < N\_int; j++) {

if (intervals[j] < intervals[i]) {

cout << "incorrect borders\n";

return 0;

}

}

}

auto numbers = new int[N];

random\_device rd;

mt19937 generator(rd());

normal\_distribution<> dist((X\_max + X\_min) / 2);

int i = 0;

while (i < N) {

double curr = dist(generator);

if (curr >= X\_min && curr <= X\_max) {

numbers[i] = int(curr);

i++;

}

}

for (int j = 0; j < N; j++) {

cout << numbers[j] << ' ';

}

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\tNumbers` count";

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

func PROC C intervals: dword, N\_int: dword, N: dword, numbers: dword, final\_answer: dword

push eax

push ebx

push ecx

push edi

push esi

mov esi, numbers

mov edi, final\_answer

mov eax, 0

checking\_loop:

mov ebx, 0

iter:

cmp ebx, N\_int

jge out\_cur\_iter

mov ecx, [esi + 4\*eax]

mov edi, intervals

cmp ecx, [edi+4\*ebx]

jl out\_cur\_iter

inc ebx

jmp iter

out\_cur\_iter:

dec ebx

mov edi, final\_answer

mov ecx, [edi+4\*ebx]

inc ecx

mov [edi+4\*ebx], ecx

next\_number:

inc eax

cmp eax, N

jg exit

jmp checking\_loop

exit:

pop edx

pop ecx

pop ebx

pop eax

pop edi

pop esi

ret

func ENDP

END