**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. 0382 |  | Андрющенко К.С. |
| Преподаватель |  | Ефремов М.А. |

Санкт-Петербург

2021

**Цель работы.**

Изучить работу с организацией связи Ассемблера с ЯВУ. Написать программу построения частотного распределения попаданий псевдослучайных целых чисел в интервалы, определённые индивидуальным заданием.

**Индивидуальное задание**.

Вариант 2.





Dx = Xmax - Xmin ; Lg1, Lgi –первая или любая левая граница; ПГпосл – правая граница последнего интервала.

1 Условие:

* Вид распределения псевдослучайных чисел: нормальное (гаусовское);

2 Условие:

Количеством ассемблерных модулей, формирующих требуемое распределение:

* 1 модуль, он сразу формирует распределение по заданным интервалам и возвращает его в головную программу, написанную на ЯВУ;

Это распределение возвращается в головную программу и выдается как основной результат в виде текстового файла.

3 Условие:

* Число интервалов может быть больше-равно (Nint ≥ Dx),
* Число интервалов не может быть меньше (Nint < Dx) диапазона изменения входных чисел;

4 Условие:

* Какие-то левые границы могут быть меньше Xmin (Lgi ≤ Xmin);
* Первая левая граница быть больше Xmin (Lg1 > Xmin);

5 Условие:

* Правая граница последнего интервала может быть больше Xmax (ПГпосл > Xmax);
* Правая граница последнего интервала не может быть больше Xmax (ПГпосл <= Xmax);

**Замечания:**

1) На ЯВУ следует реализовать только ввод исходных данных (возможно с контролем), вывод и генерацию псевдослучайных целых чисел. Всю остальную функциональность следует программировать на ассемблере.

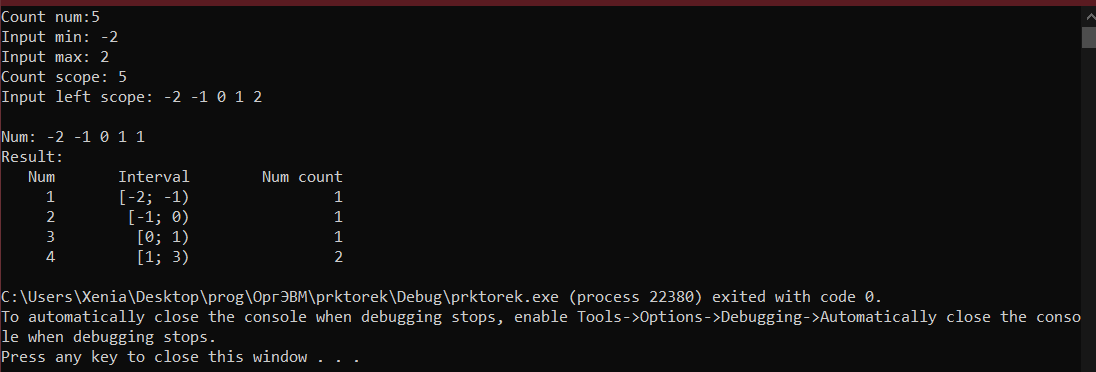
2) В отладочной версии программы (при небольшом количестве псевдослучайных чисел, не превышающем 100 значений) для контроля работы датчика сгенерированные числа, приведенные к целому виду, следует выводить на экран или в файл. В основной версии программы, предоставляемой для защиты, вывод сгенерированных псевдослучайных чисел выполнять не нужно.

**Ход работы.**

В ходе работы была разработана программа на языке Assembler и C++, которая обрабатывает построение частотного распределения попаданий псевдослучайных целых чисел в заданные интервалы в соответствие с индивидуальным заданием.

На ЯВУ считываются входные данные: кол-во генерируемых чисел, границы распределения, кол-во интервалов и сами интервалы. По условию задания кол-во интервалов >= диапазона чисел, но реализация при обратном условии не меняется. Далее высчитываются математическое ожидание и среднеквадратическое отклонения для гауссовского распределения, после чего генерируются сами числа. Затем вызывается функция из ассемблерного модуля, подсчитывающий кол-во вхождений в каждый интервал. Результат выводится в виде таблицы на экран и в файл.

Модуль содержит одну функцию, принимающую массив чисел и его размер, массив левых границ интервалов и его размер и массив для вывода. Для каждого элемента происходит поиск интервала, в который он входит, а затем кол-во вхождений для этого интервала увеличивается на единицу. По условию Lg1 > Xmin, поэтому проверяется ситуация, когда число меньше крайней левой границы, и в этом случае не учитывается. По другому условию ПГпосл > Xmax, поэтому все числа будут меньше последней границы, поэтому ситуацию, когда число выходит за крайнюю правую границу, проверять не следует. **Тестирование программы.**



Распределение чисел действительно происходит согласно гауссовскому распределению.

**Выводы.**

В ходе данной лабораторной работы была изучена организация связи Ассемблера с ЯВУ. Эти знания были применены для написания программы построения частотного распределение попаданий псевдослучайных целых чисел в заданные интервалы.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: lb6.asm

#include <iostream>

#include <iomanip>

#include <string>

#include <fstream>

#include <random>

using namespace std;

extern "C" void func(int\* nums, int numsCount, int\* leftBorders, int\* result);

void output(string A, string B, string C, ofstream& file) {

cout << setw(6) << right << A << setw(15) << right << B << setw(17) << right << C << setw(17) << right << endl;

file << setw(6) << right << A << setw(15) << right << B << setw(17) << right << C << setw(17) << right << endl;

}

int main() {

int CountNum;

cout << "Count num:";

cin >> CountNum;

if (CountNum <= 0) {

cout << "Error" << endl;

return -1;

};

int maxGateNum, minGateNum;

cout << "Input min: ";

cin >> minGateNum;

cout << "Input max: ";

cin >> maxGateNum;

if (maxGateNum <= minGateNum) {

cout << "Error" << endl;

return -1;

};

int scopeCount;

cout << "Count scope: ";

cin >> scopeCount;

cout << "Input left scope: ";

int\* leftScope = new int[scopeCount];

int\* result = new int[scopeCount];

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

cin >> leftScope[i];

int index = i;

while (index && leftScope[index] < leftScope[index - 1]) {

swap(leftScope[index--], leftScope[index]);

}

result[i] = 0;

if (i == scopeCount - 1) leftScope[i]++;

}

cout << endl;

random\_device rd{};

mt19937 gen(rd());

float mean = float(maxGateNum + minGateNum) / 2;

float stddev = float(maxGateNum - minGateNum) / 6;

normal\_distribution<float> dist(mean, stddev);

int\* num = new int[CountNum];

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

num[i] = round(dist(gen));

int index = i;

while (index && num[index] < num[index - 1]) {

swap(num[index--], num[index]);

}

}

cout << "Num: ";

for (int i = 0; i < CountNum; i++) cout << num[i] << " ";

cout << endl;

func(num, CountNum, leftScope, result);

ofstream file("output.txt");

cout << "Result:\n";

output("Num", "Interval", "Num count", file);

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

output(

to\_string(i + 1),

'[' + to\_string(leftScope[i]) + "; " + to\_string(leftScope[i + 1]) + ")",

to\_string(result[i + 1]),

file

);

}

file.close();

return 0;

}

ПРИЛОЖЕНИЕ Б

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: Source.asm

.586

.MODEL FLAT, C

.CODE

func PROC C nums:dword, numsCount:dword, leftBorders:dword, result:dword

push eax

push ebx

push ecx

push edx

push esi

push edi

mov ecx, numsCount

mov esi, nums

mov edi, leftBorders

mov edx, 0; index of current number

l:

mov ebx, [esi+4\*edx]; current number

cmp ebx, [edi]; most left border

jl continue; if x < most left border

mov eax, 0; index of interval

searchInterval:

cmp ebx, [edi+4\*eax]

jl endSearch

inc eax

jmp searchInterval

endSearch:

mov edi, result

mov ebx, [edi+4\*eax]; interval in result array

inc ebx

mov [edi+4\*eax], ebx

mov edi, leftBorders

continue:

inc edx

loop l

pop edi

pop esi

pop edx

pop ecx

pop ebx

pop eax

ret

func ENDP

END