**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

отчет

**по лабораторной работе №6**

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

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

|  |  |  |
| --- | --- | --- |
| Студент гр. 1303 |  | Жилин И.А. |
| Преподаватель |  | Ефремов М.А. |

Санкт-Петербург

2022

## Цель работы.

Изучить детали организации связи Ассемблера с ЯВУ, написать ассемблерный модуль для использования в программе.

## 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. Следует привести числа к целому виду с учетом диапазона изменения.

Далее должна вызываться ассемблерная процедура для формирования распределения количества попаданий псевдослучайных целых чисел в заданные интервалы. Ассемблерная процедура должна вызываться как независимо скомпилированный модуль. 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.

Исходные данные:

1. Длина массива псевдослучайных целых чисел - NumRanDat (<= 16K)

2. Диапазон изменения массива псевдослучайных целых чисел [Xmin, Xmax]

3. Массив псевдослучайных целых чисел {Xi}

4. Количество интервалов, на которые разбивается диапазон изменения массива псевдослучайных целых чисел - NInt (<=24)

5. Массив левых границ интервалов разбиения LGrInt

Результаты:

Текстовая таблица, строка которой содержит:

- номер интервала,

- левую границу интервала,

- количество псевдослучайных чисел, попавших в интервал.

Количество строк должно быть равно числу интервалов разбиения.

Таблица должна выводиться на экран и сохраняться в файле.

## Выполнение работы.

На языке C++ было реализовано считывание исходных данных, числа хранятся в массиве *nums*, левые границы и правая граница последнего интервала хранятся в *intervals*. Здесь же генерируется необходимое количество псевдослучайных чисел в соответствии с нормальным распределением с дисперсией, равной 2.

Изображение выглядит как текст

Автоматически созданное описание

*Рисунок 1 – Тестирование программы в консоли*

Изображение выглядит как текст

Автоматически созданное описание

*Рисунок 2 – Результат работы программы в файле*

Исходный код программы см. в приложении А.

## Вывод.

В ходе выполнения лабораторной работы были изучены детали организации связи Ассемблера с ЯВУ, написан ассемблерный модуль для использования в программе.

## ПРИЛОЖЕНИЕ А

## Тексты исходных файлов программ.

## ConsoleApplication.cpp

#include <iostream>

#include <random>

#include <fstream>

using namespace std;

extern "C" {

void func(int n, int nInt, int\* nums, int\* intervals, int\* res);

}

int main() {

int n, xMin, xMax, nInt;

cin >> n;

if (n <= 0 || n > 16000) {

cout << "Entered array length is wrong" << endl;

exit(-1);

}

cin >> xMin >> xMax;

if (xMin >= xMax) {

cout << "Entered limits are wrong" << endl;

exit(-1);

}

cin >> nInt;

if (nInt <= 0 || nInt > 24) {

cout << "Entered number of intervals is wrong" << endl;

exit(-1);

}

int \*nums = new int[n];

int\* intervals = new int[nInt + 1];

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

cin >> intervals[i];

bool wrongValue = false;

if (intervals[i] <= xMin || intervals[i] >= xMax) {

wrongValue = true;

}

if (i > 0) {

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

wrongValue = true;

}

}

if (wrongValue) {

cout << "Entered border is wrong" << endl;

delete[] nums;

delete[] intervals;

exit(-1);

}

}

cin >> intervals[nInt];

if (intervals[nInt] > xMax) {

cout << "Entered border is wrong" << endl;

delete[] nums;

delete[] intervals;

exit(-1);

}

random\_device rd;

mt19937 gen(rd());

normal\_distribution<> d(0, 2);

int it = 0;

while (it < n) {

double value = round(d(gen));

if (value >= xMin && value <= xMax) {

nums[it] = int(value);

it++;

}

}

fstream outFile;

outFile.open("res.txt", ios::out | ios::trunc);

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

cout << nums[i] << ' ';

outFile << nums[i] << ' ';

}

cout << endl;

outFile << endl;

int\* res = new int[nInt]{0};

func(n, nInt, nums, intervals, res);

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

if (i < nInt - 1) {

cout << "Interval number: " << i + 1 << "; left border: " << intervals[i] << "; numbers quantity: " << res[i] << endl;

outFile << "Interval number: " << i + 1 << "; left border: " << intervals[i] << "; numbers quantity: " << res[i] << endl;

} else {

cout << "Interval number: " << i + 1 << "; left border: " << intervals[i] << "; last right border: " << intervals[nInt] << "; numbers quantity: " << res[i] << endl;

outFile << "Interval number: " << i + 1 << "; left border: " << intervals[i] << "; last right border: " << intervals[nInt] << "; numbers quantity: " << res[i] << endl;

}

}

outFile.close();

delete[] nums;

delete[] intervals;

delete[] res;

return 0;

}

## Source.asm

.MODEL FLAT, C

.CODE

func PROC C n: dword, nInt: dword, nums: dword, intervals: dword, res: dword

push esi

push edi

push eax

push ebx

push ecx

push edx

mov ecx, n

mov esi, nums

mov edi, intervals

mov eax, 0

mov edx, nInt

inc edx

cycle:

mov ebx, 0

check:

cmp ebx, edx

jge next

push eax

mov eax, [esi + eax \* 4]

cmp eax, [edi + ebx \* 4]

pop eax

jl right\_num

inc ebx

jmp check

right\_num:

dec ebx

cmp ebx, -1

je next

mov esi, res

push eax

mov eax, [esi + ebx \* 4]

inc eax

mov [esi + ebx \* 4], eax

pop eax

mov esi, nums

next:

inc eax

loop cycle

finish:

pop edx

pop ecx

pop ebx

pop eax

pop edi

pop esi

ret

func ENDP

END