**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра математического обеспечения и применения ЭВМ**

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. 8381 |  | Киреев К.А. |
| Преподаватель |  | Кирьянчиков В.А. |

Санкт-Петербург

2019

**Цель работы.**

Получить практические навыки программирования на языке Ассемблера. Разработать программу на ЯВУ с использованием языка Ассемблера.

**Задание.**

На языке высокого уровня (Pascal или С) генерируется массив псевдослучайных целых чисел, изменяющихся в заданном диапазоне и имеющих равномерное распределение.

Далее должен вызываться ассемблерный модуль 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 целыхчисел - 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)

Подпрограмма формирования распределения количества попаданий псевдослучайных целых чисел в заданные интервалы реализуется в виде двух ассемблерных модулей, первый из которых формирует распределение исходных чисел по интервалам единичной длины и возвращает его в вызывающую программу на ЯВУ как промежуточный результат. Это распределение должно выводиться в текстовом виде для контроля. Затем вызывается второй ассемблерный модуль, который по этому промежуточному распределению формирует окончательное распределение псевдослучайных целых чисел по интервалам произвольной длины (с заданными границами). Это распределение возвращается в головную программу и выдается как основной результат в виде текстового файла и, возможно, графика.

**Вывод.**

В ходе выполнения данной лабораторной работы была написана программа на языке Ассемблера, которая строит частотное распределение попаданий псевдослучайных чисел в заданные интервалы. В результате выполнения лабораторной работы были получены практические навыки программирования на языке Ассемблер.

# **ПРИЛОЖЕНИЕ А**

**ТЕСТИРОВАНИЕ ПРОГРАММЫ**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Длина массива  псевдослучайных  чисел | Диапазон  изменения массива | Массив псевдослучайных чисел | Массив левых границ | Результирующий массив |
| 20 | [-10; 20] | 1 значение: 8  2 значение: 10  3 значение: 15  4 значение: 15  5 значение: 9  6 значение: 12  7 значение: -10  8 значение: -4  9 значение: -2  10 значение: 13  11 значение: 4  12 значение: 1  13 значение: -8  14 значение: -7  15 значение: -1  16 значение: 15  17 значение: 6  18 значение: 18  19 значение: 6  20 значение: 17 | 1 граница: -10  2 граница: 3  3 граница: 5  4 граница: 8  5 граница: 10 | [-10; 2] : 7  [3; 4] : 1  [5; 7] : 2  [8; 9] : 2  [10; 20] : 8 |
| 10 | [-15; -5] | 1 значение: -7  2 значение: -8  3 значение: -9  4 значение: -14  5 значение: -6  6 значение: -7  7 значение: -5  8 значение: -14  9 значение: -8  10 значение: -5 | 1 граница: -15  2 граница: -9 | [-15;-10] : 2  [-9;-5] : 8 |
| 5 | [-100; 100] | 1 значение: -13  2 значение: 6  3 значение: -2  4 значение: 14  5 значение: 40 | 1 граница: -100  2 граница: 0 | [-100;-1] : 2  [0;100] : 3 |

# **ПРИЛОЖЕНИЕ Б**

**ИСХОДНЫЙ КОД ПРОГРАММЫ. LR6.CPP**

#include <iostream>

#include <ctime>

#include <windows.h>

#include <clocale>

using namespace std;

extern "C"

{

void numel(int\*, int, int\*, int);

void processing(int\*, int\*, int\*, int, int);

}

void print(int\* arr, int NInt);

void table(int\* arr, int\* L\_arr, int NInt);

void table2(int\* arr, int\* L\_arr, int NInt);

int main()

{

srand(time(0));

setlocale(0, "");

SetConsoleOutputCP(1251);

SetConsoleCP(1251);

int NumRanDat, Xmin, Xmax, NInt;

cout << "Введите длину массива псевдослучайных чисел" << endl;

cin >> NumRanDat;

if (NumRanDat > 16 \* 1024 || NumRanDat < 1)

{ cout << "!Неверная длина" << endl; return 0; }

cout << "Введите диапазон изменения массива псевдослучайных целых чисел" << endl;

cout << "Минимум: ";

cin >> Xmin;

cout << "Максимум: ";

cin >> Xmax;

if (Xmax < Xmin)

{ cout << "!Неверный диапазон" << endl; return 0; }

int n = Xmax - Xmin + 1; // длина диапазона

int\* arr = new int[NumRanDat]; // массив псевдослучайных чисел

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

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

cout << endl << "Исходный массив:" << endl;

print(arr, NumRanDat);

int\* result = new int[n]; // массив полученного распределения едичных отрезков

int\* L = new int[n + 1]; // массив левых границ для единичных отрезков

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

{ result[i] = 0; L[i] = Xmin + i; }

numel(arr, Xmin, result, NumRanDat);

table(result, L, n);

cout << endl << "Введите количество интервалов разбиения" << endl;

cin >> NInt;

if (NInt > 24 || NInt < 1 || NInt > n)

{ cout << "!Неверное количество интервалов" << endl; return 0; }

int\* L\_array = new int[NInt + 1]; // массив левых границ

L\_array[0] = Xmin;

cout << "1 граница: " << Xmin << endl;

for (int j = 1; j < NInt; j++) {

cout << j + 1 << " граница: ";

cin >> L\_array[j];

if ((L\_array[j] < Xmin) || (L\_array[j] > Xmax))

{ cout << "!Неверная граница" << endl; return 0; }

}

cout << endl << "Массив левых границ" << endl;

print(L\_array, NInt);

int\* res = new int[NInt]; // конечный массив

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

res[i] = 0;

L\_array[NInt] = Xmax + 1;

processing(result, res, L\_array, Xmin, Xmax);

table2(res, L\_array, NInt);

system("pause");

return 0;

}

void print(int\* arr, int NInt)

{

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

cout << i + 1 << " значение: " << arr[i] << endl;

}

void table(int\* arr, int\* L\_arr, int NInt)

{

cout << endl;

cout << "№ Отрезок Количество чисел" << endl;

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

cout << j + 1 << " [" << L\_arr[j] << "]\t " << arr[j] << endl;

}

void table2(int\* arr, int\* L\_arr, int NInt)

{

cout << endl;

cout << "№ Интервал Количество чисел" << endl;

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

cout << j + 1 << " [" << L\_arr[j] << ";" << L\_arr[j + 1] - 1 << "]\t\t" << arr[j] << endl;

}

# **ПРИЛОЖЕНИЕ В**

**ИСХОДНЫЙ КОД ПРОГРАММЫ. NUMEL.ASM**

.586

.MODEL FLAT, C

.CODE

numel PROC C arr:dword, Xmin:dword, result\_array:dword, length\_:dword

mov eax, arr

mov ebx, result\_array

mov ecx, length\_

xor edx, edx

xor edi, edi

cycle :

mov edi, [eax + 4 \* edx] ; значение

sub edi, Xmin

inc dword ptr [ebx + 4 \* edi] ; увеличение счетчика числа

inc edx ; следующее значение

loop CYCLE

finish:

ret

numel endp

end

# **ПРИЛОЖЕНИЕ Г**

**ИСХОДНЫЙ КОД ПРОГРАММЫ. PROCESSING.ASM**

.586

.MODEL FLAT, C

.CODE

processing PROC C arr:dword, result:dword, L\_array:dword, Xmin:dword, Xmax:dword

;arr - массив количества всех цифр на единичном отрезке

;result - конечный массива

;L\_array - массив левых границ

;Xmin - начальное значение отрезка

;Xmax - конечное значение отрезка

mov eax, Xmin ; значение на отрезке

xor ebx, ebx

xor ecx, ecx

mov edi, result

mov edx, L\_array

mov esi, arr

cycle :

cmp eax, [edx + 4] ; если точка вышла за границы первого отрезка

jge adding ; переход на новый отрезок

mov ebx, [esi + ecx \* 4] ; количество цифр

add[edi], ebx ; добавление к отрезку

inc ecx ; счетчик для массива количества цифр

inc eax ; следующая цифра

cmp eax, Xmax

jle cycle

jmp finish

adding :

add edi, 4 ; следующая ячейка конечного массива

add edx, 4 ; следующая граница

jmp cycle

finish :

ret

processing endp

end