**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. 6304 |  | Путьков Д.А. |
| Преподаватель |  | Кирьянчиков В.А. |

Санкт-Петербург

2017

**Цель работы.**

Создание ассемблерных процедур в виде отдельных модулей и последующий их вызов из языка высокого уровня. Изучение связи языка высокого уровня C++ с Ассемблером.

# **Задание.**

На языке высокого уровня программируется ввод с клавиатуры и контроль исходных данных, а также генерируется массив псевдослучайных целых чисел, изменяющихся в заданном диапазоне и имеющих равномерное распределение.

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.

Исходные данные:

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

2. Диапазон изменения массива псевдослучайных целых чисел(minBorder;MaxBorder);

3. Массив псевдослучайных целых чисел Rand;

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

5. Массив левых границ интервалов разбиения Left.

**Задание по варианту 19**

Для бригад с **нечетным** номером: программа формирования распределения количества попаданий псевдослучайных целых чисел в заданные интервалы реализуется в виде одного ассемблерного модуля (процедуры), сразу получающего требуемое распределение и возвращающего его в головную программу, написанную на ЯВУ.

# **Ход работы.**

1. В качестве языка высокого уровня будет использоваться С++ в среде разработки Visual Studio.
2. На языке С++ создаётся функция main, в которой производятся считывание значений из командной строки, вызов функции печати результатов и процедуры, написанной на языке Ассемблера.
3. Для печати результатов в файл и на экран создается функция showResults. Код программы на языке C++ представлен в приложении А.
4. Создаётся процедура на языке Ассемблера Intervals, в которой происходит распределение исходных чисел по заданным интервалам. Содержимое файла Intervals.asm представлено в приложении Б.
5. Программа компилируется с созданием листинг файла.

**Вывод.**

В результате выполнения данной лабораторной работы были получены навыки написания процедур на языке Ассемблера в виде отдельных модулей и вызова их из программ, написанных на языках высокого уровня с передачей параметров через стек.

**Приложение А**

**Код Source.cpp**

#include "stdafx.h"

#include <time.h>

#include <ctime>

#include <iostream>

#include <windows.h>

#include <fstream>

#DEFINE MAX\_ARRAY\_LEN 16000

#DEFINE MAX\_SPLIT\_LEN 24

using namespace std;

extern "C"

{

void Intervals(int Len\_Num , int\* Rand , int\* Left, int\* Count);

}

int compare(const void \*a, const void \*b) {

return(\*(int\*)b - \*(int\*)a);

}

void showResults(int Xmin, int \*Left, int \*Count, int Split\_Num) {

ofstream fout("results.txt");

int j = 0;

cout << "\n\nDistrbution of count of be-in random numbers" << endl;

fout << "Distrbution of count of be-in random numbers" << endl;

cout << "\nSplitting Range Count of be-in elmnts" << endl;

fout << "\nSplitting Range Count of be-in elmnts" << endl;

int k = 1;

cout << " "<<k<<" [" << Xmin << ", " << Left[0] << "] " << Count[j] << endl;

fout << " " << k << " [" << Xmin << ", " << Left[0] << "] " << Count[j] << endl;

k++;

for (int i = 0; i < Split\_Num; i++)

{

cout << " " << k << " (" << Left[i] << ", " << Left[i + 1] << "] " << Count[++j] << endl;

fout << " " << k << " (" << Left[i] << ", " << Left[i + 1] << "] " << Count[j] << endl;

k++;

}

fout.close();

}

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*Lab work 6 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "\*\*\*\*\*\*\*\*\*\*\*made by Putkov Dmitrii\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

int arrayLen;

int minBorder;

int maxBorder;

int Split\_Num;

int Count[24];

while (true) {

cout << "Write length of array of random positive numbers arrayLen <= 16000: ";

cin >> arrayLen;

if (( arrayLen > MAX\_ARRAY\_LEN) || ( arrayLen < 0)) {

cout << "Error!Try again." << endl;

continue;

}

else {

cout << "Write the range of array’s changing [minBrder, maxBorder]\n" << "Write Xmin: ";

cin >> minBorder;

while (true) {

cout << "\nWrite maxBorder: ";

cin >> maxBorder;

if (maxBorder< minBorder)

{

cout << "\nError!maxBorder must be greater than maxBorder" << endl;

cout << "\nTry again." << endl;

continue;

}

else

break;

}

}

cout << "Count of points for range’s separating - Split\_Num" << endl;

while (true) {

cout << "Write positive number Split\_Num <= 24: ";

cin >> Split\_Num;

if ((Split\_Num > MAX\_SPLIT\_LEN) || (Split\_Num < 0) || (Split\_Num >(maxBorder - minBorder))) {

cout << "Error! Try again." << endl;

continue;

}

else

break;

}

int\* Left = new int[Split\_Num+1];

memset(Left, maxBorder + 1, sizeof(Left));

memset(Count, 0, sizeof(Count));

cout << "Array of left borders of separating ranges Left\n" << "Write numbers, are in from minBorder to maxBorder in grow up order: ";

for (int i = 0; i < Split\_Num; i++) {

while (true) {

cin >> Left[i];

if ((Left[i] < minBorder) || (Left[i] >= maxBorder)) {

cout << "Error!Try again." << endl;

continue;

}

else

break;

}

}

Left[Split\_Num] = maxBorder;

srand(time(NULL));

cout << "\nFake-random array of numbers:\n";

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

{

Rand[i] = rand() % (maxBorder - minBorder + 1) + Xmin;

cout << Rand[i] << ' ';

}

qsort(Rand, arrayLen, sizeof(int), compare);

cout << "\nSorted array:\n";

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

{

cout << Rand[i] << ' ';

}

Intervals(arrayLen, Rand, Left, Count);

showResults(minBorder, Left, Count, Split\_Num);

system("pause");

return 0;

}

**Приложение Б**

**Код Intervals.asm**

.model flat, C

.code

Intervals proc C arrayLen: dword, Rand: ptr dword, Left: ptr dword, Count: ptr dword

mov ecx, arrayLen

mov edx, 0

jmp Number

Next\_Border:

inc edx

jmp Border; переход в метку с этой же границей

Number:

mov eax,Rand

sub eax, 4

mov eax, [eax+ecx\*4];

Border:

mov ebx,Left

mov ebx,[ebx+edx\*4]

cmp eax,ebx ;сравнение числа и границы,

jg Next\_Border ;прыжок на следующую границу, если больше

mov ebx,Count ;увеличение кол-ва

mov eax,[ebx+edx\*4] ;чисел на промежутке

inc eax

mov [ebx+edx\*4],eax

loop Number

endpr:

ret

Intervals endp

end