**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

отчет

**по практической работе №6**

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

# **Тема: Организация связи Ассемблера с ЯВУ**

|  |  |  |
| --- | --- | --- |
| Студент гр. 8382 |  | Мирончик П.Д. |
| Преподаватель |  | Ефремов М.А. |

Санкт-Петербург

2019

**Цель работы.**

Организация связи Ассемблера с ЯВУ на примере программы построения частотного распределение попаданий псевдослучайных целых чисел в заданные интервалы.

**Формулировка задачи.**

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

**Ход работы**

В ходе работы была реализована программа, принимающая на вход диапазон, в котором будут генерироваться случайные числа, количество случайных чисел а также множество левых границ областей, в которых будет подсчитано количество сгенерированных чисел.

На языке C++ производится считывание данных и генерация случайных чисел, далее эти данные передаются в функцию Ассемблера FILL\_MAS, которая заполняет массив xMin..xMax, элементами которого является количество сгенерированных чисел для каждого значения.

Затем полученные данные передаются в функцию COMPUTE\_LIMITS, также реализованную на языке Ассемблер. Она, в свою очередь, подсчитывает количество чисел уже в конкретных заданных лимитах.

Полученный в результате всех обработок массив, содержащий количество элементов в областях, заданных пользователем, выводится на экран.

**Выводы**

В ходе выполнения данной работы были изучены возможности взаимодействия программ Ассемблера и C++ при выносе ассемблерного кода в отдельный файл, особенности передачи параметров при вызове ассемблерных функций из высокоуровневого языка, а также более тонкая работа с массивами

**Приложение А. Код файла utils.asm**

.586

.MODEL FLAT, C

.CODE

FILL\_MAS PROC ; xMin: dword, xMax: dword, numbers: dword, nCount: dword, mas: dword

push ebp

mov ecx, 0

cycle: mov ebp, [esp+16]

mov eax, [ebp+ecx\*4]

sub eax, [esp+8]

mov ebp, [esp+24]

mov esi, [ebp+eax\*4]

inc esi

;add ebp,eax

mov [ebp+eax\*4], esi

inc ecx

cmp ecx, [esp+20]

jl cycle

pop ebp

ret

FILL\_MAS ENDP

COMPUTE\_LIMITS PROC ; xMin:dw, xMax:dw, mas:dw, limitArr:dw, outArr:dw, limitMaxIndex:dw

push ebp

mov ecx, [esp+2\*4] ; current X

mov esi, 0 ; limit counter

check\_exit:

cmp ecx, [esp+3\*4]

jg exit

check\_limit:

cmp esi, [esp+7\*4]

jge limit\_is\_normal ; check is it last limit

mov ebp, [esp+5\*4]

cmp ecx, [ebp+esi\*4] ; cmp

jl limit\_is\_normal

inc esi

jmp check\_limit

limit\_is\_normal:

mov ebp, [esp+4\*4]

sub ecx, [esp+2\*4] ; choose cur mas index

mov eax, [ebp+ecx\*4] ; mas's item

add ecx, [esp+2\*4]

mov ebp, [esp+6\*4] ; out array

add eax, [ebp+esi\*4]

mov [ebp+esi\*4], eax

inc ecx

jmp check\_exit

exit: pop ebp

ret

COMPUTE\_LIMITS ENDP

END

**Приложение Б. Код файла lr6.cpp**

#include "pch.h"

#include <iostream>

#include <cstdlib>

using namespace std;

extern "C" void FILL\_MAS(int xMin, int xMax, int \*numbers, int nCount, int \*mas);

extern "C" void COMPUTE\_LIMITS(int xMin, int xMax, int \*mas, int \*ranges, int \*out, int limitsMaxIndex);

int mRand(int xMin, int xMax) {

int mr = rand();

return (double) mr / RAND\_MAX \* (xMax - xMin) + xMin;

}

int main() {

int xMin, xMax, nCount, rCount;

cout << "Enter xMin, xMax, numbers count and ranges count divided by spaces, f.e. <-10 20 2000 4>" << endl;

cin >> xMin >> xMax >> nCount >> rCount;

int \*ranges = (int\*) calloc(rCount, sizeof(int)),

\*results = (int\*) calloc(rCount + 1, sizeof(int));

cout << "Enter " << rCount << " left edges" << endl;

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

cin >> ranges[i];

}

int \*numbers = (int\*) calloc(nCount, sizeof(int));

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

int test = mRand(xMin, xMax);

numbers[i] = test;

}

int \*mas = (int\*)calloc(xMax - xMin + 1, sizeof(int));

FILL\_MAS(xMin, xMax, numbers, nCount, mas);

COMPUTE\_LIMITS(xMin, xMax, mas, ranges, results, rCount);

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

cout << results[i] << " ";

return 0;

}