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 распределения попаданий псевдослучайных целых чисел в заданные интервалы

Студент гр. 9382 _____ Субботин М.О.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург
2020

Цель работы.

Научиться организации связи Ассемблера с ЯВУ на примере программы построения частотного распределения попаданий псевдослучайных целых чисел в заданные интервалы.

Основные теоретические положения.

На языке высокого уровня (Pascal или C) генерируется массив псевдослучайных целых чисел, изменяющихся в заданном диапазоне и имеющих равномерное распределение. Необходимые датчики псевдослучайных чисел находятся в каталоге Tasks\RAND_GEN (при его отсутствии программу датчика получить у преподавателя).

Далее должен вызываться ассемблерный модуль(модули) 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.

Ход выполнения:

В C++ вводятся и подготавливаются данные. Основную работу выполняет метод, написанный на языке Ассемблера.

В Ассемблере реализуется метод, который проходится по всему массиву чисел и определяет их в нужный интервал и увеличивает количество элементов в интервале на единицу.

Результаты работы программы выводятся в файл.

Исходный код программы:

Cpp-файл:

```
#include <iostream>
#include <fstream>
#include <ctime>
```

```
using namespace std;
int getRandomNumber(int min, int max)
{
   static const double fraction = 1.0 / (static_cast<double>(RAND_MAX) + 1.0);
   // Равномерно распределяем рандомное число в нашем диапазоне
   return static_cast<int>(rand() * fraction * (max - min + 1) + min);
}

extern "C"
{
   void MAS_INTERVAL(int array_size, int* arr, int* left_boarders, int* res_arr);
}

int main()
{
   srand(static_cast<unsigned int>(time(0)));
   system("chcp 1251 > nul");
   int array_size = 0;
   cout << "Введите длину массива: ";
   cin >> array_size;
   if (array_size > 16 * 1024) {
      cout << "Слишком много элементов!";
      return 0;
   }

   int x_min = 0;
   cout << "Введите нижний диапазон: ";
   cin >> x_min;
```

```
int x_max = 0;
cout << "Введите верхний диапазон: ";
cin >> x_max;

int intervals_number = 0;
cout << "Введите количество диапазонов(<=24): ";
cin >> intervals_number;
if (intervals_number > 24) {
   cout << "Диапазон слишком много!";
   return 0;
}

int *left_boarders = new int[intervals_number];
cout << "Введите " << intervals_number - 1 << " нижних границ интервалов ";
for (int i = 0; i < intervals_number - 1; i++) {
   cin >> left_boarders[i];
   if (left_boarders[i] > x_max || left_boarders[i] < x_min) {
      cout << "Введеное граница не входит в заданные промежутки!";
      return 0;
   }
}
left_boarders[intervals_number - 1] = x_max;

int *arr = new int[array_size];
for (int i = 0; i < array_size; i++) {
   arr[i] = getRandomNumber(x_min, x_max);
}
```

```
        int *res_arr = new int[intervals_number];
        for (int i = 0; i < intervals_number; i++) {
            res_arr[i] = 0;
        }

        MAS_INTERVAL(array_size, arr, left_boarders, res_arr);

        ofstream myfile("out.txt", std::ios::out);
        for (int i = 0; i < array_size; i++) {
            cout << arr[i] << " ";
            myfile << arr[i] << " ";
        }
        myfile << endl;

        if (myfile) {
            myfile << "N_interval\tL_borders\tN_number\n";
            for (int i = 0; i < intervals_number; i++) {
                int res = i != 0 ? left_boarders[i - 1] : x_min;
                myfile << "     " << i+1 << "\t\t     " << res << "\t\t    " << res_arr[i] <<
endl;
            }
        }
    }
```

Asm-файл:

```
.686
.MODEL FLAT, C
.STACK
.DATA
.CODE
```

```
MAS_INTERVAL PROC C array_size:dword, arr:dword, left_boarders:dword, res_arr:dword
    mov ecx, 0; счетчик для прохода по массиву чисел
    mov ebx, arr ; ebx указывает на начало массива чисел
    mov edi,left_boarders; edi указывает на начало массива левых граней
    traverse_numbers:
        mov eax,[ebx]; в eax лежит текущий элемент
        push ebx; сохраняем указатель на текущий элемент
        mov ebx,0; обнуляем указатель

    traverse_borders: ;здесь ebx - счетчик границ
        mov edx,ebx; в edx лежит текующий индекс массива границ
        shl edx,2; этот индекс умножаем на 4, т.е. каждый элемент по 4 байта
        cmp eax,[edi+edx]; сравниваем текующий элемент с текующей левой границей (left_boarders + 4*i), i -номер элемента
        jle matched_interval; если число меньше либо равно левой границе, то идем в matched_interval

        inc ebx; инкрементируем указатель
        jmp traverse_borders; т.к. наше число больше левой

    matched_interval:
        add edx,res_arr; edx - сдвиг для left_boarders, после сложения edx указывает на элемент в res_arr который нужно инкрементировать

        mov eax,[edx];достаем количество подходящей левой границы
        inc eax;прибавляем к ней единицу
        mov [edx],eax;вставляем ее обратно
```

```
        pop ebx;достаем текущий сдвиг для массива чисел

        add ebx,4; перемещаем указатель на следующий элемент массива чисел
        inc ecx; инкрементируем количество разобранных элементов
        cmp ecx,array_size; смотрим, рассмотрели ли мы все элементы
        jl traverse_numbers; если еще не все, то продолжаем
    ret
    MAS_INTERVAL ENDP
    END
```

Тестирование.

№	Входные данные	Выходные данные
1	92 -77 -43 -16 90 -23 -96 56 13 -64	N_interval L_borders N_number 1 -100 3 2 -50 4 3 50 3
2	97 56 64 53 86 29 92 11 22 48 17 61 63 56 42 64 70 47 59 46	N_interval L_borders N_number 1 0 0 2 10 8 3 50 12
3	936 199 238 830 -665 768 -729 -590 880 -671	N_interval L_borders N_number 1 -1000 0 2 -750 4 3 -500 2 4 500 0 5 750 4

Выводы.

Была изучена организация связи Ассемблера с ЯВУ на примере программы построения частотного распределения попаданий псевдослучайных целых чисел в заданные интервалы.