МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО 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.1381 _____ Сагидуллин Э.Р.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург

2022

Цель работы.

Научиться объединять язык Ассемблера и язык высокого уровня так, чтобы функции ассемблерного модуля вызывались из программы на ЯВУ. Написать программу построения частотного распределения попаданий псевдослучайных целых чисел в заданные интервалы.

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 две 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 – NumRanDat

2. Диапазон изменения массива псевдослучайных целых чисел [*XXmmiiii*, *XXmmmmmm*], 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 [*XXmmiiii*, *XXmmmmmm*]). Подпрограмма формирования распределения количества попаданий псевдослучайных целых чисел в заданные интервалы реализуется в виде одного ассемблерного модуля, сразу формирующего требуемое распределение и возвращающего его в головную программу, написанную на ЯВУ.

Выполнение работы.

В файле Source.cpp происходит считывание входных данных, генерация псевдослучайных чисел, а также вывод выходных данных. В модуле, написанном на языке Ассемблера, обрабатывается массив псевдослучайных чисел. Для этого используются инструкция loop1: пока не будут обработаны все числа из массива array, функция будет обрабатывать числа. Для каждого числа поочередно ищется соответствующий ему интервал (find_border): если текущее число больше левой границы, то берется следующая граница, пока число не будет меньше текущей границы, тогда ее интервал – предыдущая граница - переходим по метку out_of_border, где соответствующий результат увеличивается на единицу. После этого переходим к следующему элементу массива array.

Исходный код программы см. в приложении А.

Тестирование:

Для проверки работоспособности были проведены тесты.

```
Vvedite kolichestvo elementov massiva randomnih chisel: 10
Vvedite promejutok randoma
Ot: 0
Do: 10
Vvedite kolichestvo intervalov: 4
Vvedite intervali v luobom poryadke
Interval #1: 0
Interval #2: 2
Interval #3: 4
Interval #4: 6

Massiv:
| 3 | 4 | 5 | 6 6 6 7 8 9 10 |

Indeks Interval Kolichestvo
  1       0       0
  2       2       1
  3       4       2
  4       6       7
```

Выводы.

Изучены принципы организации связи Ассемблера с ЯВУ, а также разработана программа, которая строит частотное распределение попаданий псевдослучайных целых чисел в заданные интервалы

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: Source1.asm

```
.586p
.MODEL FLAT, C
.CODE
border_function PROC C USES EDI ESI, array:dword, len:dword,
LGrInt:dword, NInt:dword, answer:dword

    push eax
    push ebx
    push ecx
    push edi
    push esi

    mov ecx, len
    mov esi, array
    mov edi, LGrInt
    mov eax, 0

loop1:
    mov ebx, 0
    find_border:
         cmp ebx, NInt
         jge out_of_border

         push eax
         mov eax, [esi + 4 * eax]
         cmp eax, [edi + 4 * ebx]
         pop eax
         jl out_of_border
         inc ebx
         jmp find_border

    out_of_border:
         dec ebx

         cmp ebx, -1
         je to_next_num
         mov edi, answer
         push eax
         mov eax, [edi + 4 * ebx]
         inc eax
         mov [edi + 4 * ebx], eax
         pop eax
         mov edi, LGrInt

    to_next_num:
         inc eax
```

```
loop loop1

pop esi
pop edi
pop ecx
pop ebx
pop eax

ret

border_function ENDP
END
```

Название файла: Source.cpp

```
#include <iostream>
#include <random>
#include <stdlib.h>
#include <fstream>
#include <algorithm>

extern "C" {void border_function(int* Array, int len, int* LGrInt,
int NInt, int* answer); }

namespace {
    auto randomizer = std::mt19937(std::random_device{}());

    int rand_int(int from, int to) {
        return std::uniform_int_distribution<int>(from,
to)(randomizer);
    }

    void qsortRecursive(int* mas, int size) {
        int i = 0;
        int j = size - 1;
        int mid = mas[size / 2];
        do {
            while (mas[i] < mid) {
                i++;
            }
            while (mas[j] > mid) {
                j--;
            }
            if (i <= j) {
                int tmp = mas[i];
                mas[i] = mas[j];
                mas[j] = tmp;

                i++;
                j--;
```

```
            }
        } while (i <= j);

        if (j > 0) {
            qsortRecursive(mas, j + 1);
        }
        if (i < size) {
            qsortRecursive(&mas[i], size - i);
        }
    }

}


int main()
{
    int NumRamDat;
    int Xmin;
    int Xmax;
    int NInt;
    int* Array;
    int* LGrInt;

    std::cout << "Vvedite kolichestvo elementov massiva randomnih
chisel: ";
    std::cin >> NumRamDat;
    Array = new int[NumRamDat];

    std::cout << "Vvedite promejutok randoma\nOt: ";
    std::cin >> Xmin;
    std::cout << "Do: ";
    std::cin >> Xmax;

    if (Xmin >= Xmax) {
        std::cout << "Nepravilnoe max ;(";
        return 0;
    }

    for (int i = 0; i < NumRamDat; i++) Array[i] = rand_int(Xmin,
Xmax);


    std::cout << "Vvedite kolichestvo intervalov: ";
    std::cin >> NInt;

    if (NInt < 0 || NInt > 24) {
        std::cout << "Nepravilnoe kolichestvo ;(";
        return 0;
    }

    LGrInt = new int[NInt];
```

```
    std::cout << "Vvedite intervali v luobom poryadke\n";
    for (int i = 0; i < NInt; i++)
    {
        std::cout << "Interval" <<" #" << i + 1 << ": ";
        std::cin >> LGrInt[i];
        if (LGrInt[i] > Xmax || LGrInt[i] < Xmin) {
            std::cout << "Nepravilnoe interval";
            return 0;
        }
    }

    qsortRecursive(LGrInt, NInt);

    int* answer = new int[NInt] {0};

    border_function(Array, NumRamDat, LGrInt, NInt, answer);

    std::cout << "\n";
    std::cout << "Massiv:\n";

    qsortRecursive(Array, NumRamDat);

    int j = 0;
    int split = answer[j];
    if(NInt != 0) std::cout <<"| ";
    for (int i = 0; i < NumRamDat; i++) {
        if (i+1 < split || NInt ==0) std::cout << Array[i] << " ";
        else {
            j++;
            split += answer[j];
            std::cout << Array[i] << " | ";
        }
    }


    std::cout << "\n\n";
    std::cout << "Indeks " << "Interval " << "Kolichestvo"<<
std::endl;

    for (int i = 0; i < NInt; i++) std::cout << "  " << i + 1 << "\t
" << LGrInt[i] << "\t  " << answer[i] << '\n';

}
```