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 _____ Русинов Д.А.

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 заданных 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.

Задание.

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;

Ход работы.

В C++ происходит считывание исходных данных. Основную работу выполняет метод, написанный на языке Ассемблера.

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

Результаты работы программы выводятся в файл.

Тестирование.

Номер	Входные данные	Выходные данные		
1	10 5 5 5 27 39 28 43 45 22 29 13 13 24	N_I 1 2 3 4 5	L_B 5 10 15 20 25	N_N 0 2 0 2 6
2	5 -10 10 5 -3 -1 3 8 0	N_I 1 2 3 4 5	L_B -10 -8 -6 -4 -2	N_N 0 0 0 1 4
3	1 5 10 1 7	N_I 1	L_B 5	N_N 1

Выводы.

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

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.cpp

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


int64_t getRandomNumber(int64_t min, int64_t max)
{
    return min + rand() % (max - min);
}

extern "C" void MODULE_INTERVAL(int64_t* left_boarders, int64_t* res_ar-
ray, int64_t* array, int64_t size);

int main()
{
    int64_t size = 0;
    std::cout << "Введите длину массива: ";
    std::cin >> size;
    while (size <= 0 || size > 16 * 1024) {
        if (size > 16 * 1024) std::cout << "Слишком много элементов!
Введите длину, которая меньше или равно 16*1024\n";
        else std::cout << "Кол-во элементов должно быть > 0! Введите
заново: ";
        std::cin>>size;
    }

    int64_t Xmin = 0;
    std::cout << "Введите нижний диапазон: ";
    std::cin >> Xmin;

    int64_t Xmax = 0;
    std::cout << "Введите верхний диапазон: ";
    std::cin >> Xmax;

    int64_t countIntervals = 0;
    std::cout << "Введите количество диапазонов(<=24): ";
    std::cin >> countIntervals;
    while (countIntervals > 24) {
        std::cout << "Диапазонов слишком много! Введите количество
интервалов, которое меньше или равно 24\n";
        std::cin >> countIntervals;
    }

    auto *leftBorders = new int64_t[countIntervals];
    std::cout << "Введите " << countIntervals - 1 << " нижних границ
интервалов: ";
    for (int64_t i = 0; i < countIntervals - 1; i++) {
        std::cin >> leftBorders[i];
        while (leftBorders[i] > Xmax || leftBorders[i] < Xmin) {
```

```
            std::cout << "Введеная граница " << leftBorders[i] << " не
входит в заданные промежутки! Введите снова: ";
            std::cin >> leftBorders[i];
        }
        while (i != 0 && leftBorders[i-1] >= leftBorders[i]) {
            std::cout << "Введеная граница " << leftBorders[i] << " <=
предыдущей границе! Введите снова: ";
            std::cin >> leftBorders[i];
        }
    }
    leftBorders[countIntervals - 1] = Xmax;

    auto *numberArray = new int64_t[size];
    for (int64_t i = 0; i < size; i++) {
        numberArray[i] = getRandomNumber(Xmin, Xmax);
    }

    auto *resultArray = new int64_t[countIntervals];
    for (int64_t i = 0; i < countIntervals; i++) {
        resultArray[i] = 0;
    }

    MODULE_INTERVAL(leftBorders, resultArray, numberArray, size);

    std::ofstream file("result.txt");
    std::cout<<"Сгенерированные псевдослучайные числа: ";
    file << "Сгенерированные псевдослучайные числа: ";
    for (int64_t i = 0; i < size; i++) {
        std::cout << numberArray[i] << " ";
        file << numberArray[i] << " ";
    }
    file << "\n";
    std::cout<<"\n";
    std::cout << "\nNumer_interval\tLeft_borders\tCount_number\n";
    file << "\nNumer_interval\tLeft_borders\tN_number\n";
    for (int64_t i = 0; i < countIntervals; i++) {
        int64_t res = i != 0 ? leftBorders[i - 1] : Xmin;
        file << "    " << i + 1 << "\t\t    " << res << "\t\t   " << re-
sultArray[i] << "\n";
        std::cout << "    " << i+1 << "\t\t    " << res << "\t\t   " <<
resultArray[i] << "\n";
    }
}
```

Название файла: module.s

```
.intel_syntax noprefix
.global _MODULE_INTERVAL
.text

# 64-bit System V calling convention
# rdi
# rsi
# rdx
# rcx
# r8
# r9
```

```
# rdi -> leftBorders
# rsi -> resultArray
# rdx -> numberArray
# rcx -> size

_MODULE_INTERVAL:
    SUB rax, rax            # кол-во обработанных элементов из numberAr-
ray
    SUB rbx, rbx            # счетчик обработанных границ

CHECK_BORDERS:
    MOV r8, [rdx]
    CMP r8, [rdi+rbx]
    JLE MATCHED_BORDER

    ADD rbx, 8
    JMP CHECK_BORDERS

MATCHED_BORDER:
    MOV r8, [rsi+rbx]
    INC r8
    MOV [rsi+rbx], r8

    ADD rdx, 8
    INC rax

    SUB rbx, rbx
    CMP rax, rcx
    JL CHECK_BORDERS

RET
```