

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. 9383

Камзолов Н.А.

Преподаватель

Ефремов М.А.

Санкт-Петербург

2020

Цель работы.

Научиться организовывать связь Ассемблера и ЯВУ. Написать программу построения частотного распределения попаданий псевдослучайных целых чисел в заданные интервалы.

Текст задания(Вариант 2).

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

Подпрограмма формирования распределения количества попаданий псевдослучайных целых чисел в заданные интервалы реализуется в виде двух ассемблерных модулей, первый из которых формирует распределение исходных чисел по интервалам единичной длины и возвращает его в вызывающую программу на ЯВУ как промежуточный результат. Это распределение должно выводиться в текстовом виде для контроля. Затем вызывается второй ассемблерный модуль, который по этому промежуточному распределению формирует окончательное распределение псевдослучайных целых чисел по интервалам произвольной длины (с заданными границами). Это распределение возвращается в головную программу и выдается как основной результат в виде текстового файла и, возможно, графика.

Ход работы.

Main.cpp:

В этом файле происходит считывание исходных данных, вызов функций, обрабатывающих исходные данные, а также вывод обработанных данных. Также с помощью спецификатора extern организована связь ассемблерных модулей и C++ кода.

Module1.asm:

В этом файле с помощью одного цикла мы записываем распределение чисел по единичным отрезкам, записывая в $xMin + i$ индекс кол-во повторений i -го числа.

Module2.asm:

В этом файле формируется исходное распределение чисел по интервалам. Сначала мы увеличиваем каждую границу на $xMax$, получая массив индексов, по которым лежат левые границы в массиве, полученном на первом этапе. Затем записываем в уже исходный массив распределение.

Тестирование.

```
Input array size: 20
Input min array value: -5
Input max array value: 5
Input count of intervals: 3
Input 3 left borders: -3 0 2

Рандомные числа: -5 2 3 2 -1 0 1 -5 -3 5 -4 2 1 4 -1 5 -3 4 4 -3
Распределение по единичным интервалам: 2 1 3 0 2 1 2 3 1 3 2
```

Номер интервала	Левая граница интервала	Кол-во чисел в интервале
0	-5	3
1	-3	5
2	0	3
3	2	9

Рисунок 1 – тестирование программы 1.

```

Input array size: 40
Input min array value: -20
Input max array value: 20
Input count of intervals: 4
Input 4 left borders: -15 0 5 10

Рандомные числа: 13 2 6 18 -13 16 4 -17 -2 14 -14 2 -17 4 5 -15 -6 3 7 -6 -18 11 -2 -16 -17 5 4 -6 -20 20 -18 -17 -11 -1
0 -17 1 3 -1 15 16
Распределение по единичным интервалам: 1 0 2 5 1 1 1 1 0 1 1 0 0 0 3 0 0 0 2 1 0 1 2 2 3 2 1 1 0 0 0 1 0 1 1 1 2 0 1 0 1

Номер интервала      Левая граница интервала      Кол-во чисел в интервале
0                      -20                          9
1                      -15                          11
2                      0                            8
3                      5                            4
4                      10                           8

```

Рисунок 2 – Тестирование программы 2.

Выводы.

Получены знания об организации связи Ассемблера и ЯВУ. Написана программа построения частотного распределения попаданий псевдослучайных целых чисел в заданные интервалы.

ПРИЛОЖЕНИЕ А

КОД ПРОГРАММЫ

Название файла: main.cpp

```
#include <iostream>
#include <random>
#include <time.h>
#include <fstream>

extern "C"
{
    void MODULE1(int* resArray, int* arr, int size, int xMin);
    void MODULE2(int* resArray, int resSize, int* leftBoards,
int intervalsCount, int xMin, int xMax, int* finalArray);
}

int RandomRandom(int min, int max)
{

    return min + rand() % (max - min + 1);
}

int main()
{
    setlocale(0, "");
    srand(time(NULL));
    int arraySize;
    std::cout << "Input array size: ";
    std::cin >> arraySize;
    if (arraySize > 16 * 1024)
    {
        std::cout << "Array size should be < 18*1024\n";
        return 0;
    }
}
```

```

int xMin, xMax;
std::cout << "Input min array value: ";
std::cin >> xMin;
std::cout << "Input max array value: ";
std::cin >> xMax;
if (xMin > xMax)
{
    std::cout << "Max array value should be greater than
min array value\n";
    return 0;
}
int intervalsCount;
std::cout << "Input count of intervals: ";
std::cin >> intervalsCount;
if (intervalsCount > 24)
{
    std::cout << "Count of interval should be less than
25\n";
    return 0;
}
int* leftBorders = new int[intervalsCount];
std::cout << "Input " << intervalsCount << " left borders:
";

for (int i = 0; i < intervalsCount; i++)
{
    std::cin >> leftBorders[i];
    if (leftBorders[i] > xMax || leftBorders[i] < xMin)
{
        std::cout << "Border is out of range!";
        return 0;
    }
}

int* array = new int[arraySize];

```

```

for (int i = 0; i < arraySize; i++)
{
    array[i] = RandomRandom(xMin, xMax);
}

int* res1 = new int[abs(xMax-xMin) + 1];
for (int i = 0; i < abs(xMax - xMin) + 1; i++)
{
    res1[i] = 0;
}

MODULE1(res1, array, arraySize, xMin);
std::cout << std::endl;
int* finalArray = new int[intervalsCount + 1];
for (int i = 0; i < intervalsCount + 1; i++) {
    finalArray[i] = 0;
}

MODULE2(res1, arraySize, leftBorders, intervalsCount,
xMin, xMax, finalArray);

std::ofstream textFile;
textFile.open("answer.txt");
std::cout << "Рандомные числа: ";
textFile << "Рандомные числа: ";
for (int i = 0; i < arraySize; i++)
{
    std::cout << array[i] << " ";
    textFile << array[i] << " ";
}
std::cout << std::endl;
textFile << std::endl;
std::cout << "Распределение по единичным интервалам: ";
textFile << "Распределение по единичным интервалам: ";
for (int i = 0; i < abs(xMax - xMin) + 1; i++)

```

```

        {
            std::cout << res1[i] << " ";
            textFile << res1[i] << " ";
        }
        std::cout << std::endl;
        textFile << std::endl;
        std::cout << "Номер интервала          " << "Левая граница
интервала          " << "Кол-во чисел в интервале\n";
        textFile << "Номер интервала          " << "Левая граница
интервала          " << "Кол-во чисел в интервале\n";
        std::cout << "          " << 0 << "          \t \t          " << xMin <<
"          \t\t          " << finalArray[0] << std::endl;
        textFile << "          " << 0 << "          \t\t\t\t          " << xMin <<
"          \t\t\t\t          " << finalArray[0] << std::endl;
        for (int i = 1; i < intervalsCount + 1; i++)
        {
            std::cout << "          " << i << "          \t \t          " <<
leftBorders[i-1] - xMax << "          \t\t          " << finalArray[i]
<< std::endl;
            textFile << "          " << i << "          \t\t\t\t          " <<
leftBorders[i - 1] - xMax << "          \t\t\t\t          " <<
finalArray[i] << std::endl;
        }

    }
}

```

Module1.asm:

```

.586p
.MODEL FLAT, C
.CODE
PUBLIC C MODULE1
MODULE1 PROC C RESARRAY:DWORD, ARRAY:DWORD, SIZEARR:DWORD,
XMIN:DWORD
    PUSH ESI
    PUSH EDI;сохранение регистров

```



```

        PUSH EBP

        MOV EDI, RESARRAY
        MOV ESI, ARRAY
        MOV EAX, XMIN
        MOV ECX, SIZEARR
for_loop:
        MOV EBX, [ESI]
        SUB EBX, EAX
        MOV EBP, [EDI+4*EBX]
        INC EBP
        MOV [EDI+4*EBX], EBP
        ADD ESI, 4
        LOOP for_loop

        POP EBP
        POP EDI
        POP ESI

ret
MODULE1 ENDP
ND

```

Module2.asm:

```

.586p
.MODEL FLAT, C
.CODE
PUBLIC C MODULE2
MODULE2 PROC C ARRAY:DWORD, SIZEARRAY:DWORD, BOARDERS:DWORD,
BOARDERSIZE:DWORD, XMIN:DWORD, XMAX:DWORD, RESARRAY:DWORD
        PUSH ESI
        PUSH EDI;сохранение регистров
        PUSH EBP

```

```

MOV ESI, BOARDERS ;указатель на массив границ
MOV EAX, 0 ;индекс результирующего массива
MOV EDX, XMIN ;минимальное значение
MOV ECX, BOARDERSIZE ;кол-во границ
for_loop:
    MOV EAX, [ESI]
    ADD EAX, XMAX
    MOV [ESI], EAX
    ADD ESI, 4
    loop for_loop

MOV EDI, ARRAY ;указатель на исходный массив
MOV ECX, BOARDERSIZE
MOV ESI, BOARDERS
SUB EBX, EBX
MOV EAX, [ESI]

for_loop2:
    PUSH ECX
    MOV ECX, EAX
    PUSH ESI
    MOV ESI, RESARRAY
    forik:
        MOV EAX, [EDI]
        ADD [ESI + EBX*4], EAX
        ADD EDI, 4
        loop forik
    POP ESI

MOV EAX, [ESI]
ADD ESI, 4
SUB EAX, [ESI]
NEG EAX

```

```

    INC EBX
    POP ECX
    loop for_loop2

    MOV ESI, RESARRAY
    MOV ECX, BOARDERSIZE
    SUB EAX, EAX

last_forik:
    ADD EAX, [ESI]
    ADD ESI, 4
    loop last_forik

    MOV ESI, RESARRAY
    SUB EAX, SIZEARRAY
    NEG EAX
    ADD [ESI + 4 * EBX], EAX

    POP EBP
    POP EDI
    POP ESI

    ret
MODULE2 ENDP
END

```