

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 распределения попаданий
псевдослучайных целых чисел в заданные интервалы.

Студент гр. 0383

Куликов А. В.

Преподаватель

Ефремов М. А.

Санкт-Петербург

2021

Цель работы.

Реализовать программу построения частотного распределения попаданий псевдослучайных целых чисел в заданные интервалы, связав модуль на Ассемблере с файлом на ЯВУ.

Задание.

На языке С программируется ввод с клавиатуры и контроль исходных данных, а также генерируется массив псевдослучайных целых чисел, изменяющихся в заданном диапазоне и имеющих заданный закон распределения. Необходимые датчики псевдослучайных чисел находятся в каталоге RAND_GEN (при его отсутствии получить у преподавателя).

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.

Исходные данные.

1. Длина массива псевдослучайных целых чисел - NumRanDat ($\leq 16K, K=1024$)
2. Диапазон изменения массива псевдослучайных целых чисел $[X_{\min}, X_{\max}]$, значения 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 $[X_{\min}, X_{\max}]$).

Результаты:

1. Текстовый файл, строка которого содержит:

- номер интервала,
- левую границу интервала,
- количество псевдослучайных чисел, попавших в интервал.
- Количество строк равно числу интервалов разбиения.

2. График, отражающий распределение чисел по интервалам.

(необязательный результат)

Выполнение работы.

На языке C++ реализовано считывание начальных данных (количество чисел, минимальное и максимальное значения, количество интервалов и их левые границы). Левые границы заносятся в массив `left_borders`, а генерируемые числа добавляются в массив `arr`. Отдельно создается массив, который будет хранить результат работы.

В ассемблерный модуль в процедуру `FUNC` передаются указатель на массив сгенерированных чисел, его размер, указатель на массив левых границ интервалов и его размер, указатель на массив, хранящий результат работы. В процедуре для каждого числа находится интервал, в который он попадает, и результат записывается в результирующий массив. После этого результат работы выводится в консоль и записывается в файл `out.txt`.

Разработанный программный код см. в приложении А.

Тестирование.

Табл.1 Проверка работы программы

Входные данные	Результат работы программы			Примечание
Amount: 10 Min: -4 Max: 7 Int num: 14	Generated numbers: 4 -3 -1 2 -2 3 -1 6 6 -3 Interval index Interval left border Amount of numbers in interval 1 -7 0 2 -3 3 3 -1 2			Верно

Borders: -7 - 3 -1 0 2 4 6 8 10 12 14 16 18 20	4 5 6 7 8 9 10 11 12 13 14	0 2 4 6 8 10 12 14 16 18 20	0 2 1 2 0 0 0 0 0 0 0	
Amount: 25 Min: -100 Max: 100 Int num: 4 Borders: -30 2 3 48	Generated numbers: 1 -32 29 4 -80 -20 -83 38 81 -62 15 -56 -49 64 -62 82 82 -61 -10 82 -31 53 57 38 97 Interval index Interval left border Amount of numbers in interval 1 -30 3 2 2 0 3 3 5 4 48 8			Верно

Выводы.

В ходе выполнения данной лабораторной работы была изучена организация связи модулей на языке ассемблер с ЯВУ. Была реализована программа построения частотного распределения попаданий псевдослучайных целых чисел в заданные интервалы.

ПРИЛОЖЕНИЕ А

Тексты исходных файлов программ

main.cpp

```
#include <iostream>
#include <fstream>
#include <random>

extern "C" void FUNC(int* array, int array_size, int* left_boarders, int intervals_size, int*
result_array);

int main()
{
    std::ofstream file("out.txt");
    int NumRanDat;

    std::cout << "Enter the amount of numbers" << std::endl;
    std::cin >> NumRanDat;

    int Xmin, Xmax;
    std::cout << "Enter the minimum number" << std::endl;
    std::cin >> Xmin;
    std::cout << "Enter the maximum number" << std::endl;
    std::cin >> Xmax;

    if (Xmax < Xmin)
    {
        std::cout << "Error: wrong minimum or maximum number" << std::endl;
        return 0;
    }

    int interval_amount;
    std::cout << "Enter the amount of intervals" << std::endl;
    std::cin >> interval_amount;

    if (interval_amount <= 0)
    {
        std::cout << "Error: the amount of intervals should be positive" << std::endl;
        return 0;
    }

    int* left_boarders = new int[interval_amount];
    std::cout << "Enter left borders" << std::endl;
    for (int i = 0; i < interval_amount; i++)
        std::cin >> left_boarders[i];

    for (int i = 0; i < interval_amount - 1; i++)
    {
        for (int j = i + 1; j < interval_amount; j++)
        {
            if (left_boarders[j] < left_boarders[i])
                std::swap(left_boarders[j], left_boarders[i]);
        }
    }
}
```

```

    }
}

std::random_device rand;
std::mt19937 gen(rand());
std::uniform_int_distribution<> dis(Xmin, Xmax);
int* arr = new int[NumRanDat];

for (int i = 0; i < NumRanDat; i++)
    arr[i] = dis(gen);

file << "Generated numbers: ";
for (int i = 0; i < NumRanDat; i++)
    file << arr[i] << ' ';
file << '\n';

int* result_array = new int[interval_amount];
for (int i = 0; i < interval_amount; i++)
    result_array[i] = 0;

FUNC(arr, NumRanDat, left_borders, interval_amount, result_array);

std::cout << "Interval index \tInterval left border \tAmount of numbers in interval" <<
'\n';
file << "Interval index \tInterval left border \tAmount of numbers in interval" << '\n';

for (int i = 0; i < interval_amount; i++)
{
    std::cout << "\t" << i + 1 << "\t\t" << left_borders[i] << "\t\t\t" << result_array[i]
<< '\n';
    file << "\t" << i + 1 << "\t\t" << left_borders[i] << "\t\t\t" << result_array[i] <<
'\n';
}

delete[] left_borders;
delete[] arr;
delete[] result_array;
}

```

module.asm

```

.586
.MODEL FLAT, C
.CODE
FUNC PROC C array:dword, array_size:dword, left_borders:dword, interval_amount:dword,
result_array:dword
push ecx
push esi
push edi
push eax
push ebx

```

```
mov ecx, array_size
mov esi, array
mov edi, left_borders
mov eax, 0
```

```
l1:
```

```
    mov ebx, 0
    borders:
        cmp ebx, interval_amount
        jge borders_exit
        push eax
        mov eax, [esi+4*eax]
        cmp eax, [edi+4*ebx]
        pop eax
        jl borders_exit
        inc ebx
        jmp borders
```

```
    borders_exit:
    dec ebx
```

```
    cmp ebx, -1
    je skip
    mov edi, result_array
    push eax
    mov eax, [edi+4*ebx]
    inc eax
    mov [edi+4*ebx], eax
    pop eax
    mov edi, left_borders
    skip:
    inc eax
```

```
loop l1
```

```
pop ebx
pop eax
pop edi
pop esi
pop ecx
ret
FUNC ENDP
END
```