

**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»**

**Тема: Организация связи Ассемблера с ЯВУ на примере программы  
построения частотного распределение попаданий псевдослучайных  
целых чисел в заданные интервалы.**

Студент гр. 9383

Чумак М.А.

Преподаватель

Ефремов М.А.

Санкт-Петербург

2020

### **Цель работы.**

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

### **Задание.**

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.

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;

14

3. Количество интервалов, на которые разбивается диапазон изменения массива псевдослучайных целых чисел - NInt ( $\leq 24$ )

4. Массив левых границ интервалов разбиения LGrInt (должны принадлежать интервалу  $[X_{\min}, X_{\max}]$ ).

Результаты:

2

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)

В зависимости от номера бригады формирование частотного распределения должно производиться по одному из двух вариантов:

1. Для бригад с нечетным номером: 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 главную

программу, написанную на ЯВУ;

### **Выполнение работы.**

Первоначально у пользователя запрашиваются все необходимые данные:

длина массива, нижняя и верхняя границы значений, количество интервалов,

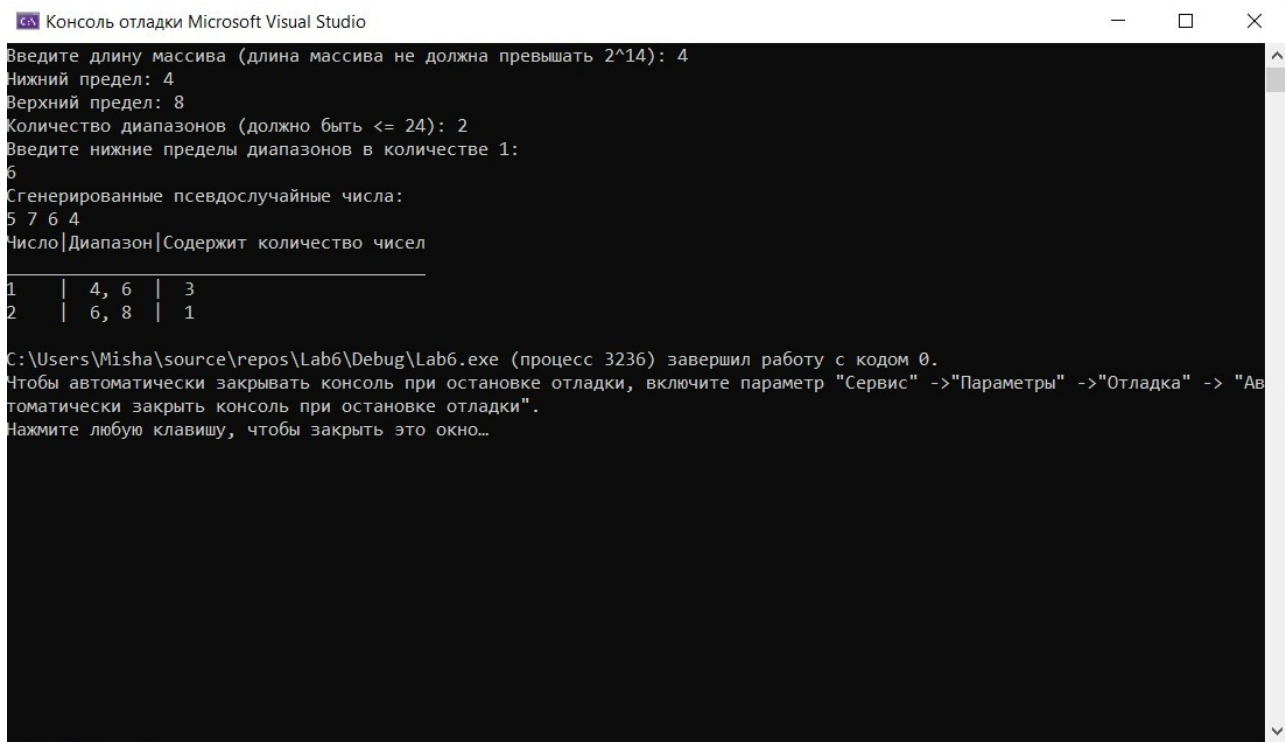
нижние границы интервалов. Затем генерируется массив псевдослучайных

чисел и передается в модуль asm для дальнейшей обработки. В модуле

заполняется массив с количеством чисел в каждом диапазоне. После этого данные выводятся пользователю, программа завершается.

Исходный код программы см. в приложении А

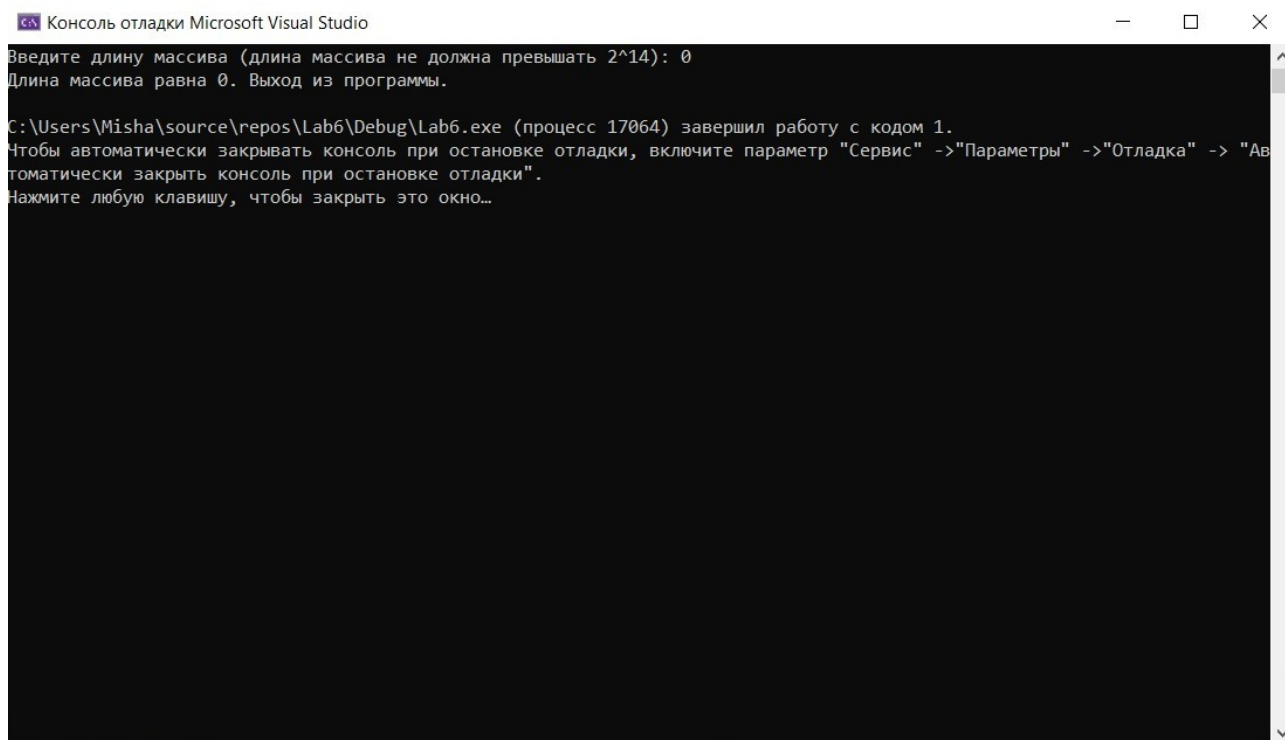
## Тестирование.



Консоль отладки Microsoft Visual Studio

```
Введите длину массива (длина массива не должна превышать 2^14): 4
Нижний предел: 4
Верхний предел: 8
Количество диапазонов (должно быть <= 24): 2
Введите нижние пределы диапазонов в количестве 1:
6
Сгенерированные псевдослучайные числа:
5 7 6 4
Число|Диапазон|Содержит количество чисел
-----
1 | 4, 6 | 3
2 | 6, 8 | 1
C:\Users\Misha\source\repos\Lab6\Debug\Lab6.exe (процесс 3236) завершил работу с кодом 0.
Чтобы автоматически закрывать консоль при остановке отладки, включите параметр "Сервис" ->"Параметры" ->"Отладка" -> "Автоматически закрыть консоль при остановке отладки".
Нажмите любую клавишу, чтобы закрыть это окно...
```

Рисунок 1 - Пример вывода программы №1



Консоль отладки Microsoft Visual Studio

```
Введите длину массива (длина массива не должна превышать 2^14): 0
Длина массива равна 0. Выход из программы.
C:\Users\Misha\source\repos\Lab6\Debug\Lab6.exe (процесс 17064) завершил работу с кодом 1.
Чтобы автоматически закрывать консоль при остановке отладки, включите параметр "Сервис" ->"Параметры" ->"Отладка" -> "Автоматически закрыть консоль при остановке отладки".
Нажмите любую клавишу, чтобы закрыть это окно...
```

Рисунок 2 - Пример вывода программы №2

```
Консоль отладки Microsoft Visual Studio
Введите длину массива (длина массива не должна превышать 2^14): 4
Нижний предел: 0
Верхний предел: -2
Верхний предел меньше нижнего. Выход из программы.

C:\Users\Misha\source\repos\Lab6\Debug\Lab6.exe (процесс 2496) завершил работу с кодом 1.
Чтобы автоматически закрывать консоль при остановке отладки, включите параметр "Сервис" ->"Параметры" ->"Отладка" -> "Автоматически закрыть консоль при остановке отладки".
Нажмите любую клавишу, чтобы закрыть это окно...
```

Рисунок 3 - Пример вывода программы №3

```
Консоль отладки Microsoft Visual Studio
Введите длину массива (длина массива не должна превышать 2^14): 4
Нижний предел: 0
Верхний предел: 5
Количество диапазонов (должно быть <= 24): -2
Неверно введено количество диапазонов. Выход из программы.

C:\Users\Misha\source\repos\Lab6\Debug\Lab6.exe (процесс 7916) завершил работу с кодом 1.
Чтобы автоматически закрывать консоль при остановке отладки, включите параметр "Сервис" ->"Параметры" ->"Отладка" -> "Автоматически закрыть консоль при остановке отладки".
Нажмите любую клавишу, чтобы закрыть это окно...
```

Рисунок 4 - Пример вывода программы №4

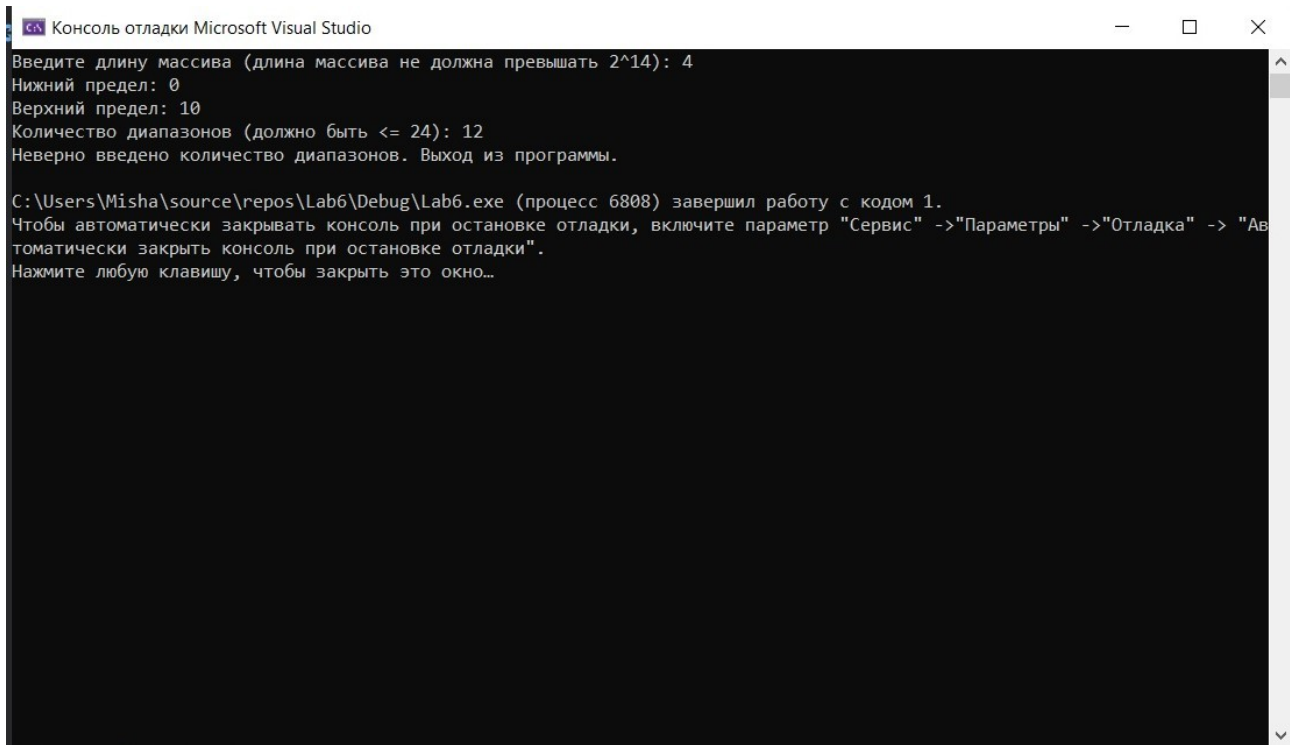
The image shows a screenshot of the 'Консоль отладки Microsoft Visual Studio' (Microsoft Visual Studio Debug Console) window. The window has a title bar with the Visual Studio logo and standard minimize, maximize, and close buttons. The console text is as follows:  
Введите длину массива (длина массива не должна превышать 2^14): 4  
Нижний предел: 0  
Верхний предел: 10  
Количество диапазонов (должно быть <= 24): 12  
Неверно введено количество диапазонов. Выход из программы.  
  
C:\Users\Misha\source\repos\Lab6\Debug\Lab6.exe (процесс 6808) завершил работу с кодом 1.  
Чтобы автоматически закрывать консоль при остановке отладки, включите параметр "Сервис" ->"Параметры" ->"Отладка" -> "Автоматически закрыть консоль при остановке отладки".  
Нажмите любую клавишу, чтобы закрыть это окно...  
The text is displayed on a black background with white font. There is a vertical scrollbar on the right side of the console area.

Рисунок 5 - Пример вывода программы №5

### **Выводы.**

В ходе работы была изучена организация связи Ассемблера с ЯВУ и написана программа построения частотного распределение попаданий псевдослучайных целых чисел в заданные интервалы.

## ПРИЛОЖЕНИЕ А ИСХОДНЫЙ КОД ПРОГРАММ

### Файл lab6.cpp

```
#include <cstdlib>
#include <iostream>
#include <fstream>
using namespace std;

extern "C" {
    void distribution(int arr_length, int* arr, int* lower_ranges_arr, int*
range_arr);
}

int main() {
    setlocale(0, "RU");
    int arr_length = 0, x_min = 0, x_max = 0, range_count = 0;
    cout << "Введите длину массива (длина массива не должна превышать
2^14): ";
    cin >> arr_length;
    if (arr_length > 16 * 1024 || arr_length < 0) {
        cout << "Неверно введена длина массива. Выход из программы." <<
endl;
        exit(1);
    }
    if (arr_length == 0) {
        cout << "Длина массива равна 0. Выход из программы." << endl;
        exit(1);
    }
    cout << "Нижний предел: ";
    cin >> x_min;
    cout << "Верхний предел: ";
```

```

cin >> x_max;
if (x_min > x_max) {
    cout << "Верхний предел меньше нижнего. Выход из программы." <<
endl;

    exit(1);
}
cout << "Количество диапазонов (должно быть <= 24): ";
cin >> range_count;
if (range_count > 24 || range_count < 1 || range_count > x_max - x_min + 1)
{
    cout << "Неверно введено количество диапазонов. Выход из
программы." << endl;
    exit(1);
}
int* lower_ranges_arr = new int[range_count];
cout << "Введите нижние пределы диапазонов в количестве " <<
range_count - 1 << ": " << endl;
for (int i = 0; i < range_count - 1; i++) {
    cin >> lower_ranges_arr[i];
    if (lower_ranges_arr[i] < lower_ranges_arr[i - 1]) {
        cout << "Введенный предел " << lower_ranges_arr[i] << " больше
предыдущего." << endl;
        cin >> lower_ranges_arr[i];
    }
    if (lower_ranges_arr[i] < x_min || lower_ranges_arr[i] > x_max) {
        cout << "Неверно задан нижний предел. Выход из программы." <<
endl;
        exit(1);
    }
}
}

```



```

lower_ranges_arr[range_count - 1] = x_max;
int* arr = new int[arr_length]();
for (int i = 0; i < arr_length; i++)
    arr[i] = x_min + rand() % (x_max - x_min);
int* range_arr = new int[range_count];
for (int i = 0; i < range_count; i++)
    range_arr[i] = 0;
distribution(arr_length, arr, lower_ranges_arr, range_arr);
ofstream output("output.txt");
cout << "Сгенерированные псевдослучайные числа:" << endl;
output << "Сгенерированные псевдослучайные числа:" << endl;
for (int i = 0; i < arr_length; i++) {
    cout << arr[i] << " ";
    output << arr[i] << " ";
}
cout << endl;
output << endl;
cout << "Число|Диапазон|Содержит количество чисел" << endl;
output << "Число|Диапазон|Содержит количество чисел" << endl;
cout << "_____ " << endl;
output << "_____ " << endl;
for (int i = 0; i < range_count; i++) {
    int count1, count2;
    if (i != 0)
        count1 = lower_ranges_arr[i - 1];
    else
        count1 = x_min;
    if (i != range_count)
        count2 = lower_ranges_arr[i];
    else

```

```

        count2 = x_max;
        output << i + 1 << "    | " << count1 << ", " << count2 << " | " <<
range_arr[i] << endl;
        cout << i + 1 << "    | " << count1 << ", " << count2 << " | " <<
range_arr[i] << endl;
    }
    delete[] lower_ranges_arr;
    delete[] arr;
    delete[] range_arr;
    return 0;
}

```

### **Файл Distribution.asm**

```

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

distribution PROC C arr_length:dword, arr:dword, lower_ranges_arr:dword,
range_arr:dword

```

```

        mov ecx, 0                ;счетчик для прохода по массиву
        mov ebx, [arr]
        mov esi, [lower_ranges_arr]
        mov edi, [range_arr]

f1:
        mov edx, [ebx]            ;берем элемент входного массива
        push ebx                  ;сохраняем указатель на текущий элемент
        sub ebx, ebx              ;обнуляем указатель

```

f2:

```
    mov eax, ebx          ;eax содержит текущий индекс массива
    shl eax, 2            ;индекс умножаем на 4, так как каждый
    cmp edx, [esi+eax]    ;сравниваем текущий элемент с текущей
    jle fe               ;левой границей
    inc ebx
    jmp f2
```

fe:

```
    add eax, edi          ;после сложения указываем на элемент в
    mov edx, [eax]        ;результатирующем массиве для инкрементирования
    inc edx
    mov [eax], edx
    pop ebx               ;забираем текущий элемент и
    add ebx, 4            ;ссылаемся на новый
    inc ecx               ;инкрементируем индекс массива
    cmp ecx, arr_length
    jl f1
```

ret

distribution ENDP

END

**Файл Distribution.lst**

Microsoft (R) Macro Assembler Version 14.28.29335.0 12/09/20 22:08:11

```

                                .686
                                .MODEL FLAT, C
                                .STACK
00000000                        .DATA
00000000                        .CODE
00000000                        distribution PROC C arr_length:dword, arr:dword,
lower_ranges_arr:dword, range_arr:dword

                                00000003 B9 00000000                mov ecx, 0                ;счетчик
для прохода по массиву
                                00000008 8B 5D 0C                mov ebx, [arr]
                                0000000B 8B 75 10                mov esi, [lower_ranges_arr]
                                0000000E 8B 7D 14                mov edi, [range_arr]

                                00000011                f1:
                                00000011 8B 13                mov edx, [ebx]                ;берем элемент
входного массива
                                00000013 53                push ebx                ;сохраняем
указатель на текущий элемент
                                00000014 2B DB                sub ebx, ebx                ;обнуляем указатель

                                00000016                f2:
                                00000016 8B C3                mov eax, ebx                ;eax содержит
текущий индекс массива границ
                                00000018 C1 E0 02                shl eax, 2                ;индекс
умножаем на 4, так как каждый элемент по 4 байт

```

```

0000001B 3B 14 06          cmp edx, [esi+eax]      ;сравниваем
текущий элемент с текущей левой границей
0000001E 7E 03          jle fe
00000020 43          inc ebx
00000021 EB F3          jmp f2

00000023          fe:
00000023 03 C7          add eax, edi      ;после сложения
указываем на элемент в результирующем массиве для инкрементирования
00000025 8B 10          mov edx, [eax]
00000027 42          inc edx
00000028 89 10          mov [eax], edx
0000002A 5B          pop ebx
;забираем текущий элемент и ссылаемся на новый
0000002B 83 C3 04          add ebx, 4
0000002E 41          inc ecx
;инкрементируем индекс массива
0000002F 3B 4D 08          cmp ecx, arr_length
00000032 7C DD          jl f1

```

ret

```

00000036          distribution ENDP

```

END

Microsoft (R) Macro Assembler Version 14.28.29335.0 12/09/20 22:08:11

Distribution.asm

Symbols 2 - 1

Segments and Groups:

N a m e	Size	Length	Align	Combine	Class
---------	------	--------	-------	---------	-------

FLAT .....	GROUP		
STACK .....	32 Bit 00000400 Para	Stack	'STACK'
_DATA .....	32 Bit 00000000 Para	Public	'DATA'
_TEXT .....	32 Bit 00000036 Para	Public	'CODE'

Procedures, parameters, and locals:

N a m e	Type	Value	Attr
distribution .....	P Near	00000000 _TEXTLength=	00000036
Public C			
arr_length .....	DWord	bp + 00000008	
arr .....	DWord	bp + 0000000C	
lower_ranges_arr .....	DWord	bp + 00000010	
range_arr .....	DWord	bp + 00000014	
f1 .....	L Near	00000011 _TEXT	
f2 .....	L Near	00000016 _TEXT	
fe .....	L Near	00000023 _TEXT	

Symbols:

N a m e	Type	Value	Attr
@CodeSize .....	Number	00000000h	
@DataSize .....	Number	00000000h	
@Interface .....	Number	00000001h	
@Model .....	Number	00000007h	
@code .....	Text	_TEXT	

@data .....	Text	FLAT
@fardata? .....	Text	FLAT
@fardata .....	Text	FLAT
@stack .....	Text	FLAT

0 Warnings

0 Errors