

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. 1381

Хомутильников Н.А.

Преподаватель

Ефремов М.А.

Санкт-Петербург

2022 г.

Цель работы

Получить практические навыки программирования на языке Ассемблера.
Разработать программу на ЯВУ с использованием языка Ассемблера.

Задание/краткие сведения

Организация связи Ассемблера с ЯВУ на примере программы построения частотного распределение попаданий псевдослучайных целых чисел в заданные интервалы. 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;
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).

Вариант работы №30 (для бригад с четным номером)

Вывод

Написана программа на ЯВУ, содержащая 2 ассемблерных модуля, которая строит распределение попадания псевдослучайных чисел на интервалы

Приложение А. Код программы (lab6.cpp, first.asm, second.asm)

Название файла lab6.cpp:

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

using namespace std;
ofstream outfile;

#define MAX_SIZE 1024 * 16

extern "C" {void first(int* array, int len, int Xmin, int* result);
            void second(int* array, int Xmin, int Xmax, int* LGrInt, int*
arr_out); }

void print_first(int Xmin, int Xmax, int* arr){
    cout << endl << "1st module result: " << endl;
    cout << "Interval:\t" << "Value:\t" << "Count:" << endl;
    for (int i = Xmin, j = 0; i <= Xmax; i++, j++) {
        cout << "    " << j + 1 << "\t" << "    " << i << "\t" <<
arr[j] << endl;
    }
}

void print_second(int NInt, int NumRanDat, int*& arr, int*& LGrInt, int*& answer)
{
    cout << "2nd module result:" << endl;
    outfile << endl;
    cout << "Number\t" << "Left value\t" << "Count" << endl;
    outfile << "Number\t" << "Left value\t" << "Count" << endl;
    for (int i = 0; i < NInt; i++) {
        cout << "    " << i + 1 << "\t" << "    " << LGrInt[i] << "\t" <<
answer[i] << endl;
        outfile << "    " << i + 1 << "\t" << "    " << LGrInt[i] << "\t" <<
<< answer[i] << endl;
    }
}

int comparator(const void* a, const void* b) {
    return *(int*)a - *(int*)b;
}

int main(){
    srand(time(nullptr));

    int NumRanDat, Xmin, Xmax, NInt;
    cout << "Enter array length: ";
    cin >> NumRanDat;
    while (NumRanDat <= 0 || NumRanDat > MAX_SIZE){
        cout << "Size is incorrect, try again:";
        cin >> NumRanDat;
    }

    cout << "Enter borders of pseudonumeric array:" << endl;
    cout << "Min: ";
    cin >> Xmin;
    cout << "Max: ";
```

```

cin >> Xmax;
if (Xmax < Xmin)
{
    swap(Xmin, Xmax);
    cout << "Min is greater than max. Swapping..." << endl;
}

int rangeLen = Xmax - Xmin + 1;
int* arr = new int[NumRanDat];
for (int i = 0; i < NumRanDat; i++)
    arr[i] = Xmin + rand() % rangeLen;

cout << "Current array: ";
for (int i = 0; i < NumRanDat; i++)
    cout << arr[i] << " ";
cout << endl;

qsort(arr, NumRanDat, sizeof(int), comparator);

cout << "Sorted array: ";
for (int i = 0; i < NumRanDat; i++)
    cout << arr[i] << " ";

int* first_arr = new int[rangeLen] {0};
first(arr, NumRanDat, Xmin, first_arr);
print_first(Xmin, Xmax, first_arr);

cout << endl << "Input number of left borders: ";
cin >> NInt;
while (NInt <= 0 || NInt > 24 || NInt > rangeLen)
{
    cout << "Incorrect value. Try again: ";
    cin >> NInt;
}

int* LGrInt = new int[NInt + 1];
cout << "Enter value of left borders: " << endl;
for (int i = 0; i < NInt; i++)
{
    std::cin >> LGrInt[i];
    while (LGrInt[i] > Xmax || LGrInt[i] < Xmin) {
        std::cout << "Value is incorrect. Try again: ";
        std::cin >> LGrInt[i];
    }
}
LGrInt[NInt] = Xmax + 1;

int* second_arr = new int[NInt] {0};
second(first_arr, Xmin, Xmax, LGrInt, second_arr);
outfile.open("answer.txt", ios_base::out);
print_second(NInt, NumRanDat, arr, LGrInt, second_arr);
delete[] arr;
delete[] LGrInt;
delete[] first_arr;
delete[] second_arr;
outfile.close();
}

```

Название файла first.asm:

```
.586
.MODEL FLAT, C
.CODE
first PROC C arr:dword, NumRanDat:dword, Xmin:dword, answer_arr:dword

mov eax, arr
mov ebx, answer_arr
mov ecx, NumRanDat
xor edx, edx
xor edi, edi

finding:
    mov edi, [eax + 4 * edx]
    sub edi, Xmin
    inc dword ptr [ebx + 4 * edi]
    inc edx
    loop finding

    ret
first endp
end
```

Название файла second.asm:

```
.586
.MODEL FLAT, C
.CODE
second PROC C arr:dword, Xmin:dword, Xmax:dword, LGrInt:dword,
resultArr:dword

mov eax, Xmin
mov edi, resultArr
mov edx, LGrInt
mov esi, arr
xor ebx, ebx
xor ecx, ecx

finding:
    cmp eax, [edx + 4]
    jl adding
    cmp eax, Xmax
    jg finish
    cmp eax, Xmax
    je func

    add edx, 4
    add edi, 4
    jmp finding

func:
    add edi, 4
    mov ebx, [esi + ecx * 4]
    add [edi], ebx
    jmp finish

adding:
    mov ebx, [esi + ecx * 4]
    add [edi], ebx
    inc ecx
    inc eax
    jmp finding
```

```
finish:
    ret
second endp
end
```

Приложение Б. Тестирование.

Результаты тестирования представлены на рисунках 1-2 + на рисунке 3 – результат записи в текстовый файл:

```
Enter array length: 20
Enter borders of pseudonumeric array:
Min: 5
Max: 7
Current array: 7 5 7 5 5 6 7 5 7 7 6 7 5 7 7 7 6 5 5 7
Sorted array: 5 5 5 5 5 5 5 6 6 6 7 7 7 7 7 7 7 7 7 7
1st module result:
Interval:      Value:  Count:
  1           5         7
  2           6         3
  3           7        10

Input number of left borders: 2
Enter value of left borders:
5 6
2nd module result:
Number  Left value  Count
  1         5         7
  2         6        13
```

Рисунок 1

```
Enter array length: 5
Enter borders of pseudonumeric array:
Min: 2
Max: 7
Current array: 6 7 4 4 5
Sorted array: 4 4 5 6 7
1st module result:
Interval:      Value:  Count:
  1           2         0
  2           3         0
  3           4         2
  4           5         1
  5           6         1
  6           7         1

Input number of left borders: 2
Enter value of left borders:
4 5
2nd module result:
Number  Left value  Count
  1         4         2
  2         5         3
```

Рисунок 2


```
answer.txt  second.asm  first.asm  lab6.cpp
1
2  Number  Left value  Count
3  |  1      4      2
4  |  2      5      3
5
```

Рисунок 3