

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 распределение попаданий
псевдослучайных целых чисел в заданные интервалы

Студент гр. 1382

_____ Исайкин Г. И.

Преподаватель

_____ Ефремов М. А
Санкт-Петербург

2022

Цель работы.

Создать программу на ЯВУ, чьи функции, выполняющие основную логику, реализованы на ассемблере.

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
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.

Текстовый файл, строка которого содержит:

- номер интервала,
- левую границу интервала,
- количество псевдослучайных чисел, попавших в интервал.

Количество строк равно числу интервалов разбиения..

Выполнение работы.

Часть на ЯВУ выполнена на языке C++ (в ОС Windows).

В части на ЯВУ выполняется ввод начальных данных, а именно количество случайных чисел, их диапазон, количество интервалов и их диапазон. После программа генерирует нужное количество случайных чисел в нужном диапазоне (для генерации чисел используется mt19937). После вызывается функция `function1`, реализованная на ассемблере. Она перебирает массив случайных чисел и считает количество одинаковых чисел, путём добавления единицы в одно из значений массива ответов, где каждая ячейка соответствует числу из диапазона. На этом `function1` заканчивается. Далее в части на ЯВУ вызывается другая функция на ассемблере `function2`. Она переберет массив, полученный при завершении работы `function1` и складывает ячейки, входящие в определённый интервал, и заносит это значение в массив ответов, в котором каждая ячейка соответствует интервалу. На этом функция завершает работу. Далее часть на ЯВУ записывает решение в файл `out.txt`. На этом программа завершает работу. Тесты работы программы представлены в табл. 1.

Выводы.

Создана программа на высоком языке C++ с реализацией основной логики в функциях, написанных на Ассемблере.

Таблица 1 — Тесты программы

№ теста	Ввод	Вывод
1	20 /кол. чисел 1 10 /диапазон 3 /кол. интервалов 1 4 7 /лев. гран.	10 3 8 6 6 5 8 2 10 10 3 8 8 7 7 2 5 1 7 8 /числа 1 2 2 0 2 2 3 5 0 3 /результат function1 (1) = 5; (4) = 4; (7) = 11 /результат function2
2	30 -5 20 4 -5 0 5 13	5 5 2 7 11 14 4 -4 -3 3 16 -3 16 3 8 20 18 -1 4 13 20 -4 6 3 9 3 -4 4 10 2 0 3 2 0 1 0 0 2 4 3 2 1 1 1 1 1 0 1 1 0 2 0 1 0 2 (-5) = 6; (0) = 9; (5) = 8; (13) = 7
3	23 178 202 7 178 180 185 189 199 201 202	191 185 201 197 193 191 178 201 195 182 180 186 188 188 189 197 194 192 178 191 183 197 184 2 0 1 0 1 1 1 1 1 0 2 1 0 3 1 1 1 1 0 3 0 0 0 2 0 (178) = 2; (180) = 4; (185) = 4; (189) = 11; (199) = 0; (201) = 2; (202) = 0

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: lab6.cpp

```
#include <iostream>
#include <fstream>
#include <random>
using namespace std;
std::ofstream file("out.txt");
extern "C" {void function1(int* array, int arr_len, int* answer,
int mi, int v); }
extern "C" {void function2(int* array, int arr_len, int*
l_borders, int bord_len, int mi, int* answer); }
void sort(int*arr, int count_) {
    for (int i = 0; i < count_ - 1; i++) {
        for (int j = 0; j < count_ - i - 1; j++) {
            if (arr[j] > arr[j + 1]) {
                int buf = arr[j];
                arr[j] = arr[j + 1];
                arr[j + 1] = buf;
            }
        }
    }
}
int main() {
    int length;
    do{
        cout << "Length\n";
        cin >> length;
    }
    while((length > 16384) || (length <= 0));
    int mi, ma;
    do{
        cout << "min\n";
        cin >> mi;
        cout << "max\n";
        cin >> ma;
    }
    while(mi > ma);
    int *arr = new int[length];
    std::mt19937 r;
    std::random_device device;
    r.seed(device());
    for(int i = 0; i < length; i++){
        arr[i] = mi + r() % (ma - mi + 1);
        cout << arr[i] << ' ';
    }
    cout << endl;
    int count;
    do{
        cout << "count\n";
        cin >> count;
    }
}
```

```

while((count > 24) || (count < 1) || (count > (ma - mi + 1)));
    int *l_borders = new int[count];
    for(int i = 0; i < count; i++){
        do{
            cout << i << "\n";
            cin >> l_borders[i];
        }
        while((l_borders[i] < mi) || (l_borders[i] > ma));
    }
    sort(l_borders, count);
    int* buf_result = new int[ma - mi + 1] {0};
    function1(arr, length, buf_result, mi, 0);
    for (int i = 0; i < ma - mi + 1; i++) {
        cout << buf_result[i] << ' ';
    }
    cout << endl;
    int* final_result = new int[count] {0};
    function2(buf_result, ma - mi + 1, l_borders, count, mi,
final_result);
    for(int i = 0; i < count; i++){
        file<< '(' << l_borders[i] << " ) = " << final_result[i] << " ";
    }
    delete[] arr;
    delete[] l_borders;
    delete[] buf_result;
    delete[] final_result;
    file.close();
    return 0;
}

```

Название файла: lab6_1.asm

```
.586p
.MODEL FLAT, C
.CODE
function1 PROC C USES EDI ESI, array:dword, arr_len:dword,
answer:dword, mi:dword, v:dword

    push eax
    push ebx
    push edx
    push esi
    push edi
    mov eax, 0
    mov ebx, 0
    mov esi, array
    mov edi, answer
count:
    cmp ebx, arr_len
    jge out1
    mov eax, [esi+ebx*4]
    mov v, eax
    sub eax, mi
    mov edx, [edi+eax*4]
    inc edx
    mov [edi+eax*4], edx
    add ebx, 1
    jmp count
out1:
    pop edi
    pop esi
    pop edx
    pop ebx
    pop eax
    ret
function1 ENDP
END
```

Название файла: lab6_2.asm

```
.586p
.MODEL FLAT, C
.CODE
function2 PROC C USES EDI ESI, array:dword, arr_len:dword,
l_borders:dword, bord_len:dword, mi:dword, answer:dword
    push eax
    push ebx
    push ecx
    push edx
    push edi
    push esi
    mov ecx, arr_len
    mov ebx, bord_len
    dec ebx
    mov eax, arr_len
    add eax, mi
    dec eax
    mov esi, array
    mov edi, l_borders
    mov edx, 0
f:
    cmp eax, [edi+ebx*4]
    jl ff
    dec ecx
    add edx, [esi+ecx*4]
    inc ecx
    dec eax
loop f
ff:
    mov edi, answer
    mov [edi+ebx*4], edx
    mov edx, 0
    mov edi, l_borders
    dec ebx
    cmp ecx, 0
jnz f
pop esi
pop edi
pop edx
pop ecx
pop ebx
pop eax
ret
function2 ENDP
END
```