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.

Студентка гр. 0382 _____ Здобнова К.Д.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург

2021

Цель работы.

Написать программу построения частного распределения попаданий псевдослучайных целых чисел в заданные интервалы.

Задание.

Вариант: 5

Вид распределения: равномерный

Число ассем. процедур: 1

Nint < Dx

Lg1 ≤ Xmin

Исходные данные:

1. Длина массива псевдослучайных целых чисел - NumRanDat (<= 16K)
2. Диапазон изменения массива псевдослучайных целых чисел

   [Xmin, Xmax] (м.б. биполярный, например, [-100, 100])
3. Массив псевдослучайных целых чисел {Xi}.
4. Количество интервалов, на которые разбивается диапазон

   изменения массива псевдослучайных целых чисел - NInt (<=24)
5. Массив левых границ интервалов разбиения LGrInt.

Результаты:

Текстовая таблица, строка которой содержит:

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

Количество строк должно быть равно числу интервалов разбиения.

Таблица должна выводиться на экран и сохраняться в файле.

Выполнение работы.

Программа написана с использованием языка программирования C++. В

файле с расширением cpp записываются исходные данные, с соблюдением требований границ интервала. В программе, написанной на ассемблере обрабатываются массив псевдослучайных чисел. Для этого используются инструкция loop: пока не будут обработаны все числа из массива numbers, функция будет обрабатывать числа. Для каждого числа поочередно ищется нужный интервал (в метке finding_border): если текущее число больше левой границы, то берется следующая граница, пока число не будет меньше границы, тогда ее интервал – предыдущая граница - переходим по метке exit, где соответствующий результат увеличивается на единицу.

Тестирование

```
Size of array:
12
X_min:
5
X_max:
25
Number of intervals, n_int < 20:
4
LG1 <= 5
2 7 15 20
10 15 20 12 13 19 15 12 24 9 13 21
0 6 3 3
N  Lg  Res
0   2    0
1   7    6
2   15    3
3   20    3
```

```
Size of array:
15
X_min:
3
X_max:
15
Number of intervals, n_int < 12:
6
LG1 <= 3
2 5 8 10 12 13
13 12 4 3 6 12 10 5 5 14 3 11 15 11 9
3 3 1 3 2 3
N  Lg  Res
0   2    3
1   5    3
2   8    1
3   10    3
4   12    2
5   13    3
```

Выводы.

Написана программа построения частного распределения попаданий псевдослучайных целых чисел в заданные интервалы.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: *lab6.cpp*

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

extern "C" void FUNC(int* numbers, int num_cnt, int *lgrint, int
n_int, int* result);

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

int main() {
    int n, x_min, x_max, n_int, d_x;
    std::cout << "Size of array:" << std::endl;
    std::cin >> n;
    std::cout << "X_min:" << std::endl;
    std::cin >> x_min;
    std::cout << "X_max:" << std::endl;
    std::cin >> x_max;
    if (x_max < x_min) {
        std::cout << "Wrong value of x_max" << std::endl;
        return 0;
    }
    d_x = x_max - x_min;
    std::cout << "Number of intervals, n_int < " << d_x << ":" <<
std::endl;
    std::cin >> n_int;
    auto lgrint = new int[n_int + 1];
    std::cout << "LG1 <= " << x_min << std::endl;
```

```
    for(int i = 0; i < n_int; i++) {
          std::cin >> lgrint[i];
          if (lgrint[0] > x_min){
                return 0;
          }
    }
    qsort(lgrint, n_int, sizeof(int*), comp);
    lgrint[n_int] = x_max;
    std::random_device rand;
    std::mt19937 gen(rand());
    std::uniform_int_distribution<> numb(x_min, x_max);
    auto numbers = new int[n];
    for (int i = 0; i < n; i++) {
          numbers[i] = numb(gen);
    }
    for (int i = 0; i < n; i++) {
          std::cout << numbers[i] << " ";
    }
    std::cout << std::endl;
    auto result = new int[n];
    for (int i = 0; i < n_int; i++) {
          result[i] = 0;
    }
    FUNC(numbers, n, lgrint, n_int, result);
    for (int i = 0; i < n_int; i++) {
          std::cout << result[i] << " ";
    }
    std::cout << std::endl;
    std::ofstream fout("fout.txt");
    std::cout << "N  Lg  Res" << std::endl;
    fout << "N   Lg   Res" << std::endl;
    for (int i = 0; i < n_int; i++) {
          std::cout << i << "   " << lgrint[i] << "    " << result[i]
<< std::endl;
          fout << i << "   " << lgrint[i] << "    " << result[i] <<
std::endl;
    }
```

```
            fout.close();
            return 0;
      }
```

Название файла: *module.asm*

```
.586
.MODEL FLAT, C
.CODE
FUNC PROC C numbers:dword, num_cnt:dword, lgrint:dword, n_int:dword,
result:dword
push esi
push edi
push eax
push ebx
push ecx

mov esi, numbers
mov edi, lgrint
mov ecx, num_cnt
mov eax, 0
loop1:
      mov ebx, 0
      finding_border:
            cmp ebx, n_int
            jge exit
            push eax
            mov eax, [esi + 4 * eax]
            cmp eax, [edi + 4 * ebx]
            pop eax
            jl exit
            inc ebx
            jmp finding_border
      exit:
      dec ebx
      mov edi, result
      push eax
      mov eax, [edi + 4 * ebx]
      inc eax
      mov [edi + 4 * ebx], eax
      pop eax
      mov edi, lgrint
      inc eax
loop loop1
```

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