

МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В. И. УЛЬЯНОВА (ЛЕНИНА)
КАФЕДРА МО ЭВМ

ОТЧЕТ

по лабораторной работе №6
по дисциплине «ОргЭВМиС»

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

Студент гр. 0382

Афанасьев Н. С.

Преподаватели

Ефремов М.А.

Санкт-Петербург

2021

Цель работы.

Изучить организацию связи Ассемблера с ЯВУ. Применить эти знания для написания программы построения частотного распределения попаданий псевдослучайных целых чисел в заданные интервалы.

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.

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.

Вариант 2:

- Нормальное распределение
- $N_{\text{int}} \geq D_x$
- $L_{g1} > X_{\text{min}}$
- $R_{g\text{посл}} > X_{\text{max}}$

Выполнение работы.

Для начала на ЯВУ считываются входные данные: кол-во генерируемых чисел, границы распределения, кол-во интервалов и сами интервалы. По условию задания кол-во интервалов \geq диапазона чисел, но реализация при обратном условии не меняется. Далее высчитываются математическое

ожидание и среднеквадратическое отклонения для гауссовского распределения, после чего генерируются сами числа. Затем вызывается функция из ассемблерного модуля, подсчитывающий кол-во вхождений в каждый интервал. Результат выводится в виде таблицы на экран и в файл.

Сам модуль содержит одну функцию, принимающую массив чисел и его размер, массив левых границ интервалов и его размер и массив для вывода. Для каждого элемента происходит поиск интервала, в который он входит, а затем кол-во вхождений для этого интервала увеличивается на единицу. По условию $Lg_1 > X_{\min}$, поэтому проверяется ситуация, когда число меньше крайней левой границы, и в этом случае не учитывается. По другому условию $ПГ_{\text{посл}} > X_{\max}$, поэтому все числа будут меньше последней границы, поэтому ситуацию, когда число выходит за крайнюю правую границу, проверять не следует.

Разработанный программный код см. в приложении А.

Тестирование.

Примеры работы программы:

Введите кол-во псевдослучайных целых чисел: 100000

Введите границы: -110 110

Введите количество интервалов: 12

Введите левые границы: -110 -90 -70 -50 -30 -10 10 30 50 70 90 110

Результат:

Номер	Интервал	Кол-во значений
1	[-110; -90)	5533
2	[-90; -70)	20571
3	[-70; -50)	56771
4	[-50; -30)	118642
5	[-30; -10)	184607
6	[-10; 10)	214863
7	[10; 30)	187079
8	[30; 50)	122125
9	[50; 70)	59411
10	[70; 90)	21823
11	[90; 110)	5825

Введите кол-во псевдослучайных целых чисел: 1000

Введите границы: -10 0

Введите количество интервалов: 11

Введите левые границы: -10 -9 -8 -7 -6 -5 -4 -3 -2 -1 0

Результат:

Номер	Интервал	Кол-во значений
1	[-10; -9)	6
2	[-9; -8)	11
3	[-8; -7)	52
4	[-7; -6)	131
5	[-6; -5)	193
6	[-5; -4)	234
7	[-4; -3)	196
8	[-3; -2)	111
9	[-2; -1)	50
10	[-1; 0)	11

Введите кол-во псевдослучайных целых чисел: 1000

Введите границы: 500 1000

Введите количество интервалов: 3

Введите левые границы: 750 1000 1010

Результат:

Номер	Интервал	Кол-во значений
1	[750; 1000)	508
2	[1000; 1010)	0

Рис.1 - Примеры

Как видно из примеров распределение чисел действительно происходит согласно гауссовскому распределению.

Выводы.

Была изучена организация связи Ассемблера с ЯВУ. Эти знания были применены для написания программы построения частотного распределение попаданий псевдослучайных целых чисел в заданные интервалы.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЕ КОДЫ ПРОГРАММ

Название файла: lr6.cpp

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

using namespace std;

extern "C" void func(int* nums, int numsCount, int* leftBorders, int*
result);

void output(string A, string B, string C, ofstream& file) {
    cout << setw(6) << right << A << setw(15) << right << B << setw(17)
<< right << C << endl;
    file << setw(6) << right << A << setw(15) << right << B << setw(17)
<< right << C << endl;
}

int main(){
    setlocale(LC_ALL, "ru");

    int randNumCount;
    cout << "Введите кол-во псевдослучайных целых чисел: ";
    cin >> randNumCount;
    if (randNumCount <= 0) { cout << "Некорректное кол-во чисел"; return
-1; };

    int max, min;
    cout << "Введите границы: ";
    cin >> min >> max;
    if (max <= min) { cout << "Некорректные границы распределения";
return -1; };

    int intervalCount;
    cout << "Введите количество интервалов: ";
    cin >> intervalCount;
    if (randNumCount <= 0) { cout << "Некорректное кол-во интервалов";
return -1; };

    cout << "Введите левые границы: ";
    int* leftBorders = new int[intervalCount];
    int* result = new int[intervalCount];
    for (int i = 0; i < intervalCount; i++) {
        cin >> leftBorders[i];

        int index = i;
        while (index && leftBorders[index] < leftBorders[index - 1]) {
            swap(leftBorders[index--], leftBorders[index]);
        }
        result[i] = 0;
    }
    cout << endl;
```

```

random_device rd{};
mt19937 gen(rd());

float mean = float(max+min)/2;
float stddev = float(max-min)/6;
normal_distribution<float> dist(mean, stddev);

int* nums = new int[randNumCount];
for (int i = 0; i < randNumCount; i++) {
    nums[i] = round(dist(gen));
}

func(nums, randNumCount, leftBorders, result);

ofstream file("output.txt");
cout << "Результат:\n";
output("Номер", "Интервал", "Кол-во значений", file);
for (int i = 0; i < intervalCount-1; i++) {
    output(
        to_string(i + 1),
        '[' + to_string(leftBorders[i]) + "; " +
to_string(leftBorders[i + 1]) + ")",
        to_string(result[i + 1]),
        file
    );
}

file.close();
system("pause");
return 0;
}

```

Название файла: module.asm

```

.586
.MODEL FLAT, C
.CODE
    func PROC C    nums:dword,    numsCount:dword,    leftBorders:dword,
result:dword
        push eax
        push ebx
        push ecx
        push edx
        push esi
        push edi

        mov ecx, numsCount
        mov esi, nums
        mov edi, leftBorders

        mov edx, 0; index of current number
l:
        mov ebx, [esi+4*edx]; current number
        cmp ebx, [edi]; most left border
        jl continue; if x < most left border

        mov eax, 0; index of interval

```

```

        searchInterval:
            cmp ebx, [edi+4*eax]
            jl endSearch
            inc eax
            jmp searchInterval
        endSearch:

        mov edi, result
        mov ebx, [edi+4*eax]; interval in result array
        inc ebx
        mov [edi+4*eax], ebx
        mov edi, leftBorders

        continue:
        inc edx
        loop 1

        pop edi
        pop esi
        pop edx
        pop ecx
        pop ebx
        pop eax
        ret
    func ENDP
END

```