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. 0383 _____ Пустовалова Е.М.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург
2021

Цель работы.

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. Необходимые датчики псевдослучайных чисел находятся в каталоге RAND_GEN (при его отсутствии получить у преподавателя).

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, сфор-мированный 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 (<= 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 .

В общем случае интервалы разбиения диапазона изменения псевдослучайных чисел могут иметь различную длину, левые границы могут задаваться в произвольном порядке и иметь произвольные значения. Если Xmin < LGrInt(1), то часть данных не будет участвовать в формировании распределения. Каждый интервал, кроме последнего, следует интерпретировать

как [LGrInt(i), LGrInt(i+1)). Если у последнего интервала правая граница меньше Xmax, то часть данных не будет участвовать в формировании распределения.

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.

Задание.

Вариант 11.

Равномерное распределение, 2 ассемблерных процедуры, Nint < Dx, Lg1 > Xmin, ПГпосл <= Xmax.

Выполнение работы.

В ходе выполнения лабораторной работы было написано 3 модуля, 1 из которых на языке C++, и 2 на ассемблере. На C++ написан lab6.cpp, через который происходит ввод данных для выполнения программы, проверка этих данных на корректность, вызов двух модулей, написанных на ассемблере, и вывод результата работы программы в консоль и файл. Первый модуль на ассемблере (lab61.asm) распределяет сгенерированный массив чисел по единичным отрезкам, то есть проходится по всем числам в цикле loop и добавляет единицу в соответствующий отрезок. Второй модуль (lab62.asm) формирует распределение на основе первого, но по заданным пользователем интервалам. Для этого сначала находится отрезок из первого модуля, который соответствует левой границе интервала, а потом происходит обход по всем элементам до правой границы (далее прибавляется 1, если он не равен 0).

Исходный код программы находится в приложении А.

Выводы.

В ходе выполнения данной лабораторной работы была реализована программа частотного распределения случайных чисел по заданным интервалам на языке C++ с использованием ассемблерных модулей.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: lab6.cpp

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

using namespace std;

extern "C" void lab61(int* numbers, int n_size, int* result, int xmin);
extern "C" void lab62(int* array, int array_size, int xmin, int*
intervals, int inter_size, int* result);


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

    srand(time(nullptr));
    ofstream result("output.txt");

    int n_size;
    cout << "Введите количество чисел:\n";
    cin >> n_size;
    if (n_size > 16 * 1024) {
        cout << "Количество чисел должно быть меньше или равно 16*1024\n";
        return 0;
    }

    int xmin, xmax;
    cout << "Введите xmin и xmax:\n";
    cin >> xmin >> xmax;
    int Dx = xmax - xmin;
    if (Dx > 24) {
        cout << "Xmax и Xmin не дожны отличаться больше, чем на 24\n";
        return 0;
    }

    int inter_size;
    cout << "Введите количество границ:\n";
    cin >> inter_size;
```

```
if (inter_size > 24) {
    cout << "Число интервалов должно быть меньше или равно 24\n";
    return 0;
}
if (inter_size >= Dx) {
    cout << "Число интервалов должно быть меньше dx\n";
    return 0;
}

int* intervals;
int* intervals2;
int* numbers;
numbers = new int[n_size];
intervals = new int[inter_size];
intervals2 = new int[inter_size];

int* result1;
int* result2;
int lenmod1 = abs(xmax - xmin) + 1;
result1 = new int[lenmod1];
for (int i = 0; i < lenmod1; i++)
    result1[i] = 0;


result2 = new int[inter_size + 1];
for (int i = 0; i < inter_size + 1; i++)
    result2[i] = 0;


cout << "Введите все границы:\n";
for (int i = 0; i < inter_size; i++) {
    cin >> intervals[i];
    if (intervals[i] < xmin) {
        cout << "Левая граница должна быть больше либо равна Xmin\n";
        return 0;
    }
    intervals2[i] = intervals[i];

}


std::random_device rd;
```

```
        std::mt19937 gen(rd());
        std::uniform_int_distribution<> dis(xmin, xmax);
        for (int i = 0; i < n_size; i++) numbers[i] = dis(gen);

        cout << "Сгенерированные значения\n";
        result << "Сгенерированные значения\n";
        for (int i = 0; i < n_size; i++) {
            cout << numbers[i] << ' ';
            result << numbers[i] << ' ';
        }
        cout << '\n';
        cout << '\n';
        result << '\n';
        result << '\n';

        lab61(numbers, n_size, result1, xmin);
        lab62(result1, n_size, xmin, intervals, inter_size, result2);

        cout << "Результат:\n";
        result << "Результат:\n";
        cout << "№\tГраница\tКоличество чисел" << endl;
        result << "№\tГраница\tКоличество чисел" << endl;

        for (int i = 1; i < inter_size + 1; i++) {
            cout << i << "\t" << intervals2[i - 1] << '\t' << result2[i] <<
endl;
            result << i << "\t" << intervals2[i - 1] << '\t' << result2[i] <<
endl;
        }

        delete[] numbers;
        delete[] intervals;
        delete[] intervals2;
        delete[] result1;
        delete[] result2;

        return 0;
    }
```

Название файла: lab61.asm

```
.586p
.MODEL FLAT, C
```

```
.CODE
PUBLIC C lab61
Lab61 PROC C array: dword, arraysize: dword, res: dword, xmin: dword

push esi
push edi

mov edi, array
mov ecx, arraysize
mov esi, res

numbers:
    mov eax, [edi]
    sub eax, xmin
    mov ebx, [esi + 4*eax]
    inc ebx
    mov [esi + 4*eax], ebx
    add edi, 4
    loop numbers

pop edi
pop esi

ret
lab61 ENDP
END
```

Название файла: lab62.asm

```
.586p
.MODEL FLAT, C
.CODE
PUBLIC C lab62
lab62 PROC C array: dword, array_size: dword, xmin: dword, borders: dword,
intN: dword, result: dword

push esi
push edi
push ebp

mov edi, array
mov esi, borders
mov ecx, intN
```

```
borders_loop:
    mov eax, [esi]
    sub eax, xmin
    mov [esi], eax
    add esi, 4
    loop borders_loop

mov esi, borders
mov ecx, intN
mov ebx, 0
mov eax, [esi]

for_start:
    push ecx
    mov ecx, eax
    push esi
    mov esi, result

    for_array:
        cmp ecx, 0
        je end_for
        mov eax, [edi]
        add [esi + 4*ebx], eax
        add edi, 4
        loop for_array

end_for:
    pop esi
    inc ebx
    mov eax, [esi]
    add esi, 4
    sub eax, [esi]
    neg eax
    pop ecx
    loop for_start

mov esi, result
mov ecx, intN
mov eax, 0
```

```
fin_loop:
    add eax, [esi]
    add esi, 4
    loop fin_loop

mov esi, result
sub eax, array_size
neg eax

add [esi + 4*ebx], eax

pop ebp
pop edi
pop esi

ret
lab62 ENDP
END
```