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»

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

Студент гр. 0383 _____ Трофимов К.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 $\{X_i\}$.
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.

Вариант 10

Распределение — нормально. Число ассемблерных процедур 1. Nint < Dx, Первая левая граница > Xmin, Правая граница последнего интервала > Xmax.

Порядок выполнения работы.

В функции main происходит считывание данных с консоли, проверка введённых значений на корректность. Проверяем длину интервала. Созданием нормального распределения происходит при помощи функции стандартной библиотеки *normal_distribution.* Полученные левые границы сортируем. Проверяем меньше ли минимальное значение самого левого интервала. В случае не корректности введённых данных пользователю выдаётся сообщение об ошибке и программа завершается с кодом 0.

Для подсчёт чисел входящих в соответствующий интервал написана функция на языке ассемблера. В нём реализована необходимая функция, принимающая на вход массив чисел, длину массива, массив границ, длину массива границ и массив куда надо записать результат работы функции — количество чисел входящих в интервал. С помощью цикла *l1* просматриваются все элементы массива, а с помощью меток *borders* и *border_exit* просматриваются все границы. После выхода из цикла проверяется был ли найден подходящий интервал. В случае, если необходимый интервал был найден, значение в *result* увеличивается на 1.

Результат работы программы записывается в файл и выводится на экран.

Вывод.

В ходе выполнения данной лабораторной работы была изучена

организация связи ассемблера с ЯВУ. Была реализована программа частотного распределения случайных чисел по заданным интервалам на языке C++ с использованием ассемблерного модуля.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММ

Файл file1.cpp

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

extern "C" void FUNC(int *array, int array_size, int *left_borders, int
intervals_size, int *result);

int main()
{

    setlocale(0, "");

    int array_size;

    std::cout << "Введите число генерируемых чисел: ";

    std::cin >> array_size;

    int xMin, xMax;

    std::cout << "Введите минимальное значение: ";

    std::cin >> xMin;

    std::cout << "Введите максимальное значение: ";

    std::cin >> xMax;

    if (xMax < xMin)
    {

        std::cout << "Неверно введены максимальное и минимальное значения";

        return 0;
    }

    int intervals_size;

    std::cout << "Введите количество интервалов: ";

    std::cin >> intervals_size;

    if (intervals_size <= 0 or intervals_size > 24)
    {

        std::cout << "Неверно введено количество интервалов";

        return 0;
    }

    if (intervals_size >= std::abs(xMax - xMin))
    {
```

```
        std::cout << "Неверно введено количество интервалов.\nКоличество интервалов должно быть меньше длины диапозона возможных значений.";

        return 0;
    }

    int *left_borders = new int[intervals_size];

    std::cout << "Введите левые границы: ";

    for (int i = 0; i < intervals_size; i++)

        std::cin >> left_borders[i];

    for (int i = 0; i < intervals_size - 1; i++)
    {

        for (int j = i + 1; j < intervals_size; j++)
        {

            if (left_borders[j] < left_borders[i])
            {

                std::swap(left_borders[j], left_borders[i]);
            }
        }
    }

    if (intervals_size > 0 and left_borders[0] < xMin)
    {

        std::cout << "Некоторые левые границы интервалов меньше минимального возомжного значения";

        return 0;
    }

    std::random_device rd;

    std::mt19937 gen(rd());

    std::normal_distribution<> dis((xMin + xMax) / 2, std::abs(xMax - xMin) / 4);

    int *array = new int[array_size];

    for (int i = 0; i < array_size; i++)
        array[i] = std::round(dis(gen));

    std::ofstream file("out.txt");

    file << "Сгенерированные числа: ";

    for (int i = 0; i < array_size; i++)
        file << array[i] << ' ';

    file << '\n';
```

```
    std::cout << "Сгенерированные числа: ";

    for (int i = 0; i < array_size; i++)
        std::cout << array[i] << ' ';

    std::cout << '\n';

    int *result = new int[intervals_size];

    for (int i = 0; i < intervals_size; i++)

        result[i] = 0;

    FUNC(array, array_size, left_borders, intervals_size, result);

    std::cout << "Номер интервала \tЛевая граница интервала \tКоличество
чисел в интервале" << '\n';

    file << "Номер интервала \tЛевая граница интервала \tКоличество чисел
в интервале" << '\n';

    for (int i = 0; i < intervals_size; i++)
    {

        std::cout << "\t" << i + 1 << "\t\t\t" << left_borders[i] <<
"\t\t\t" << result[i] << '\n';

        file << "\t" << i + 1 << "\t\t\t" << left_borders[i] << "\t\t\t"
<< result[i] << '\n';
    }

    file.close();

    system("pause");

    return 0;
}
```

Файл file2.asm

```
     .586
     .MODEL FLAT, C
     .CODE
     FUNC  PROC  C  array:dword,  array_size:dword,  left_boarders:dword,
intervals_size:dword, result:dword
     push ecx
     push esi
     push edi
     push eax
     push ebx;

     mov ecx, array_size
     mov esi, array
     mov edi, left_borders
     mov eax, 0;
     label:
          mov ebx, 0
          borders:
               cmp ebx, intervals_size ;
```

```
            jge borders_exit
            push eax
            mov eax, [esi+4*eax]
            cmp eax, [edi+4*ebx]
            pop eax
            jl borders_exit
            inc ebx
            jmp borders
        borders_exit:
        dec ebx

        cmp ebx, -1
        je skip
        mov edi, result
        push eax
        mov eax, [edi+4*ebx]
        inc eax
        mov [edi+4*ebx], eax
        pop eax
        mov edi, left_borders
        skip:
        inc eax
loop label

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