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 распределения попаданий псевдослучайных чисел в заданные интервалы»

Студент гр. 1383 _____ Петров А.С.

Преподаватель _____ Ефремов М. А.

Санкт-Петербург

2022

Цель работы.

Разработать программу, которая генерирует массив псевдослучайных чисел на ЯВУ, а действия с ним производит на языке Ассемблера.

Задание на лабораторную работу.

На языке C++ 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.

Выполнение работы.

В начале работы программы на ЯВУ считываются необходимые данные с клавиатуры, затем происходит генерация массива псевдослучайных чисел по считанным данным. Реализованы исключительные ситуации на случай ввода недопустимых значений по условию задания. С помощью функции func1 на языке Ассемблера происходит вычисления частоты встречания каждого числа в сгенерированном массиве. Массив начинается с числа – минимума в сгенерированном массиве, а заканчивается максимумом из сгенерированного массива. После вывода на экран массива полученного из функции func1, вызывается функция func2, которая рассчитывает количество чисел, попавших

в каждый диапазон. Далее на ЯВУ для каждого диапазона выводится на экран количество этих чисел.

Тестирование.

Номер теста	Входные данные	Выходные данные
1	10 1 10 2 1 6	10 10 7 7 6 6 7 10 1 3 1 0 1 0 0 2 3 0 0 3 0 1 2 1 6 8
2	6 1 20 2 1 15	16 16 19 20 20 5 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 2 0 0 1 2 0 1 1 1 15 5

Выводы.

В ходе выполнения работы была разработана программа выполняющая считывание данных и построение по данным массива псевдослучайных чисел, с обработкой массива на языке Ассемблера.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.cpp

```
#include <iostream>
#include <fstream>
#include <random>
#include <ctime>
#include <windows.h>

extern "C" void func1(int* array, int* res, int len, int min);
extern "C" void func2(int* boards, int* intervals, int*
distribution, int difference, int Nint, int X_min);

int main() {
std::cout << "Input len array\n";
int len = 0;
std::cin >> len;
if (len > 16384 || len <= 0) {
     std::cout << "error\n";
     return 0;
}
int* array = new int[len];
int X_min = 0;
int X_max = 0;
std::cout << "Input left and right border\n";
std::cin >> X_min >> X_max;
if ((X_max - X_min) <= 0) {
     std::cout << "Error input borders\n";
     return 0;
}
int Nint = 0;
std::cout << "Input number of intervals\n";
std::cin >> Nint;
if (Nint > 24 || Nint <= 0) {
     std::cout << "Error input number of intervals\n";
     return 0;
}
int* boards = new int[Nint];
std::cout << "Input boards intervals\n";
for (int i = 0; i < Nint; i++) {
     std::cin >> boards[i];
}

int difference = X_max - X_min + 1;
srand(time(NULL));
for (int i = 0; i < len; i++) {
     array[i] = X_min + rand() % (difference);
}
```

```
int* distribution = new int[difference];
for (int i = 0; i < difference; i++) {
     distribution[i] = 0;
}
func1(array, distribution, len, X_min);
std::cout << "Array\n";
for (int i = 0; i < len; i++) {
     std::cout << array[i] << ' ';
}
std::cout << '\n';
std::cout << "Distribution array\n";
for (int i = 0; i < difference; i++) {
     std::cout << distribution[i] << ' ';
}
std::cout << '\n';


int* intervals = new int[Nint];
for (int i = 0; i < 0; i++) {
     intervals[i] = 0;
}
func2(boards, intervals, distribution, difference, Nint, X_min);
std::ofstream out;
out.open("result.txt");
for (int i = 0; i < Nint; i++) {
     std::cout << i << ' ' << boards[i] << ' ' << intervals[i] <<
'\n';
     out << i << ' ' << boards[i] << ' ' << intervals[i] << '\n';
}
out.close();
delete intervals;
delete boards;
delete distribution;
delete array;
return 0;
}
```

Название файла func1.asm

```
.MODEL FLAT
.CODE
public func1
func1 proc PROC C array : dword, res : dword, len : dword, min :
dword

push eax
push ebx
push edx
push edi
push esi

mov eax, 0
mov esi, array
mov edi, res

temp:
cmp eax, len
je exit
mov ebx, [esi + 4*eax]
sub ebx, min
mov edx, [edi + 4*ebx]
inc edx
mov [edi + 4*ebx], edx
inc eax
jmp temp

exit:
pop eax
pop ebx
pop edx
pop edi
pop esi
ret
func1 ENDP
END
```

Название файла func2.asm

```
.MODEL FLAT
.CODE
public func2
func2 proc PROC C boards : dword, intervals : dword, distribution
: dword, difference : dword, Nint : dword, X_min : dword

push eax
push ebx
push ecx
push edx
push edi
push esi
mov ecx, difference
mov ebx, Nint
dec ebx
mov eax, difference
add eax, X_min
dec eax
mov esi, distribution
mov edi, boards
mov edx, 0
func:
    cmp eax, [edi+ebx*4]
    jl temp
    dec ecx
    add edx, [esi+ecx*4]
    inc ecx
    dec eax
loop func
temp:
    mov edi, intervals
    mov [edi+ebx*4], edx
    mov edx, 0
    mov edi, boards
    dec ebx
    cmp ecx, 0
    jnz func

pop esi
pop edi
pop edx
pop ecx
pop ebx
pop eax
ret
func2 ENDP
END
```