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 целых чисел в заданные интервалы.

Студент гр. 0383 _____ Сабанов П.А.

Преподаватель _____ Ефремов М.А.

Санкт-Петербург
2021

Цель работы.

На языке высокого уровня (Pascal или C) генерируется массив псевдослучайных целых чисел, изменяющихся в заданном диапазоне и имеющих равномерное распределение. Необходимые датчики псевдослучайных чисел находятся в каталоге Tasks\RAND_GEN (при его отсутствии программу датчика получить у преподавателя).

Далее должен вызываться ассемблерный модуль(модули) для формирования распределения количества попаданий псевдослучайных целых чисел в заданные интервалы. В общем случае интервалы разбиения диапазона изменения псевдослучайных чисел могут иметь различную длину.

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.

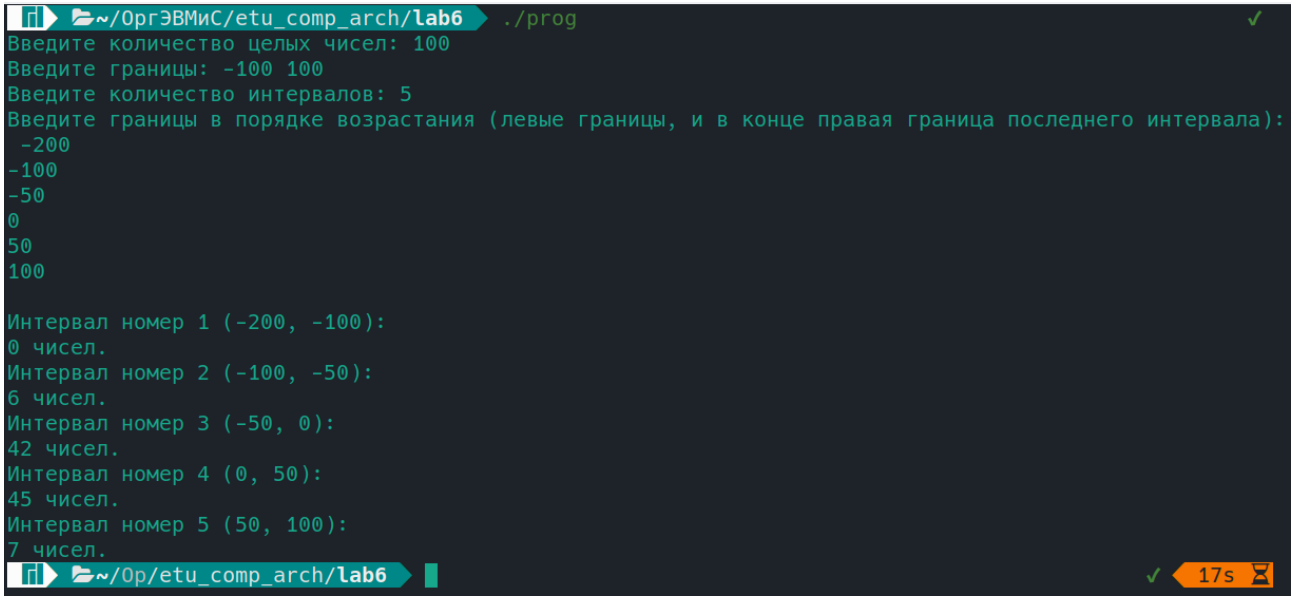
Ход работы.

Была написана функция count на языке ассемблера, принимающая пять аргументов: указатель на массив с числами, количество чисел, указатель на массив с границами интервалов, количество интервалов и указатель на массив с результатами подсчётов.

В функции count написан двойной цикл, который пробегает все интервалы и на каждом из них пробегает все числа, увеличивая счётчик

Была написана программа на языке C++, считывающая с потока стандартного вывода количество чисел, границы распределения чисел, количество и границы интервалов, вычисляющая псевдослучайные числа в нормальном распределении, передающая нужные данные функции count и выводящая содержимое массива с результатами подсчётов.

Пример работы программы.



Вывод.

Была написана программа на C++ и ассемблере nasm. Была реализована функция на ассемблере, вызываемая в программе на C++. Программа выполняет генерацию псевдослучайных чисел в произвольном диапазоне и считает количество этих чисел в произвольных интервалах.

ПРИЛОЖЕНИЕ А

Исходный код программы

main.c

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

extern "C" void count(int* nums, int numsCount, int* borders, int intervalCount, int* result);

int main() {

    int randNumCount;
    std::cout << "Введите количество целых чисел: ";
    std::cin >> randNumCount;
    while (randNumCount <= 0) {
        std::cout << "Некорректное количество чисел, попробуйте еще раз.\nВведите количество интервалов: ";
        std::cin >> randNumCount;
    }

    int min_x, max_x;
    std::cout << "Введите границы: ";
    std::cin >> min_x >> max_x;
    while (max_x <= min_x) {
        std::cout << "Некорректные границы распределения, попробуйте еще раз.\nВведите количество интервалов: ";
        std::cin >> min_x >> max_x;
    }

    int intervalCount;
    std::cout << "Введите количество интервалов: ";
    std::cin >> intervalCount;
    while (intervalCount <= 0 || max_x - min_x <= intervalCount) {
        std::cout << "Некорректное количество интервалов, попробуйте еще раз.\nВведите количество интервалов: ";
        std::cin >> intervalCount;
    }

    std::cout << "Введите границы в порядке возрастания (левые границы, и в конце правая граница последнего интервала): ";
    int* borders = new int[intervalCount + 1];
    int* result = new int[intervalCount] { 0 };
    for (int i = 0; i < intervalCount + 1; i++)
        std::cin >> borders[i];

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

    float expectation = float(max_x + min_x) / 2; // мат ожидание
    float stddev = float(max_x - min_x) / 6; // мат отклонение
```

```
    std::normal_distribution<float> dist(expectation, stddev);

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

    count(nums, randNumCount, borders, intervalCount, result);

    std::cout << '\n';
    for (int i = 0; i < intervalCount; ++i) {
        std::cout << "Интервал номер " << i+1 << " (" << borders[i] << ", " << borders[i+1] << "):\n";
        std::cout << result[i] << " чисел.\n";
    }
}
```

count.asm

```
global count

section .text

; for (int i = 0; i < intervalCount; ++i) {
;     for (int j = 0; j < numsCount; ++j) {
;         if (nums[j] >= borders[i] && nums[j] < borders[i+1]) {
;             ++result[i];
;         }
;     }
; }
count:
%define nums [ebp+8]
%define numsCount [ebp+12]
%define borders [ebp+16]
%define intervalCount [ebp+20]
%define result [ebp+24]

    push ebp
    mov ebp, esp

    push dword 0 ; int i = 0
    push dword 0 ; int j = 0

%define i [ebp-4]
%define j [ebp-8]

    push ebx

count.for_i:
```

```
; if i >= intervalCount then exit
mov eax, intervalCount
cmp i, eax
jge count.for_i.exit

mov dword j, 0 ; j = 0 before for_j

; result[i] = 0
mov edx, result
mov ecx, i
mov [edx+4*ecx], dword 0

count.for_i.for_j:

   ; if j >= numsCount then exit
   mov eax, numsCount
   cmp j, eax
   jge count.for_i.for_j.exit

   ; eax = nums[j]
   mov edx, nums
   mov ecx, j
   mov eax, [edx+4*ecx]

   ; ebx = borders[i]
   mov edx, borders
   mov ecx, i
   mov ebx, [edx+4*ecx]

   ; if nums[j] < borders[i] then next iteration
   cmp eax, ebx
   jl count.for_i.for_j.end_iteration

   ; ebx = borders[i+1]
   ;mov edx, borders ; <<-- borders already in edx
   ;mov ecx, i ; <<-- i already in ecx
   inc ecx
   mov ebx, [edx+4*ecx]

   ; if nums[j] >= borders[i] then next iteration
   cmp eax, ebx
   jge count.for_i.for_j.end_iteration

   ; ++result[i]
   mov edx, result
   ;mov ecx, i ; <<-- i+1 already in ecx
```

```
        dec ecx     ; then just decrement ecx
        inc dword [edx+4*ecx]

        count.for_i.for_j.end_iteration:

        inc dword j; ++j

        jmp count.for_i.for_j

    count.for_i.for_j.exit:

count.for_i.end_iteration:

    inc dword i ; ++i

    jmp count.for_i

count.for_i.exit:

    pop ebx
    mov esp, ebp
    pop ebp

    ret
```