

МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МОЭВМ

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.**

Студентка гр. 9382

Круглова В. Д.

Преподаватель

Ефремов М.А.

Санкт-Петербург

2020

Цель работы.

Научиться организации связи ассемблера с ЯВУ.

Задание.

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:

Данные считываются с помощью команд на ЯВУ, после чего вызывается ассемблерный модуль, который обрабатывает входной массив и возвращает готовый результат на ЯВУ.

Тестирование.

Входные данные	Выходные данные
Введите длину массива: 10 Введите нижний интервал: 1	Набор случайных чисел: 2 6 2 6 5 4 7 5 4 9 <div> <div>Номер интервала</div> <div>Левая граница</div> <div>Количество чисел</div> </div> <div> <div>1</div> <div>1</div> <div>6</div> </div>


```

int main() {
    int64_t NInt = 0;
    int64_t Xmin = 0;
    int64_t Xmax = 0;
    int64_t count = 0;
    int64_t border;

    std::cout << "Введите длину массива: ";
    std::cin >> NInt;
    // Проверка длины массива
    while (NInt > 16384) {
        std::cout << "Длина больше допустимой, введите
заново: ";
        std::cin >> NInt;
    }

    std::cout << "Введите нижний интервал: ";
    std::cin >> Xmin;

    std::cout << "Введите верхний интервал: ";
    std::cin >> Xmax;
    // Проверка верхнего интервала
    while (Xmax <= Xmin) {
        std::cout << "Введен не корректный верхний интервал,
введите еще раз: " << '\n';
        std::cin >> Xmax;
    }

    std::cout << "Введите количество интервалов: ";
    std::cin >> count;
    // Проверка интервалов
    while (count > 24) {
        std::cout << "Введено не корректное число, введите еще
раз: ";
        std::cin >> count;
    }

    int64_t *LGrInt = new int64_t[count];

```

```

        std::cout << "Введите " << count - 1 << " нижних границ
интервалов: ";
        // Считывание нижних границ
        for (int64_t i = 0; i < count - 1; i++) {
            std::cin >> LGrInt[i];
            while (LGrInt[i] > Xmax || LGrInt[i] < Xmin) {
                std::cout << "Введенная граница " << LGrInt[i] << "
не входит в заданные промежутки! Введите еще раз: ";
                std::cin >> LGrInt[i];
            }
        }

        LGrInt[count - 1] = Xmax;

        int64_t *array = new int64_t[NInt];
        // Генерация псевдослучайных чисел
        for (int64_t i = 0; i < NInt; i++) {
            array[i] = RandD(Xmin, Xmax);
        }
        // Обнуляем массив ответ
        int64_t *borderult = new int64_t[count];
        for (int64_t i = 0; i < count; i++) {
            borderult[i] = 0;
        }
        // Вызов ассемблерного модуля
        INTERVAL_SORTING(LGrInt, borderult, array, NInt);
        // Запись в файл и вывод на экран
        std::ofstream out_file("borderult.txt");
        std::cout << "Набор случайных чисел: ";
        out_file << "Набор случайных чисел: ";
        for (int64_t i = 0; i < NInt; i++) {
            std::cout << array[i] << " ";
            out_file << array[i] << " ";
        }
        out_file << "\n";
        std::cout << "\n";
        std::cout << "\nНомер интервала\tЛевая граница\tКоличество
чисел\n";
        out_file << "\nНомер интервала\tЛевая граница\tКоличество
чисел\n";

```

```

        for (int64_t i = 0; i < count; i++) {
            if(i != 0) {
                border = LGrInt[i - 1];
            }
            else {
                border = Xmin;
            }

            out_file << "          " << i+1 << "\t\t" << " <<
border << "\t\t\t" << borderult[i] << "\n";
            std::cout << "          " << i+1 << "\t" << " <<
border << "\t\t" << borderult[i] << "\n";
        }
        // Освобождение памяти
        delete borderult;
        delete array;
        delete LGrInt;

    }

```

III.s:

```

.intel_syntax noprefix
.global INTERVAL_SORTING
.text

INTERVAL_SORTING:  # rdi : LGrInt,    rsi : result ,    rdx :
array,    rcx : NInt

    WokWork:
        mov rax, [rdx]                # в rax лежит текущий
элемент
        push rdx                      # сохраняем указатель
на текущий элемент
        mov rdx, 0                    # обнуляем указатель

    Index_case:
        mov rbx, rdx                  # в rbx лежит текущий индекс
из LGrInt
        shl rbx, 3                    # этот индекс умножаем
на 8, т.е. каждый элемент по 8 байт

```

```

        cmp rax, [rdi + rbx]      # сравниваем текущий элемент
массива с текущей границей
        jge search_case          # если элемент массива
больше границы
        jmp write_case

search_case:
        inc rdx                  # для перехода к
следующему индексу массива границ
        jmp Index_case

write_case:
        add rbx, rsi             # rbx указывает на индекс
LGrInt
        mov rax, [rbx]           # rax хранит индекс LGrInt
который надо ++
        inc rax                  # увеличиваем индекс
на один
        mov [rbx], rax;          # возвращаем обратно
        pop rdx                  # восстанавливаем
указатель на array
        add rdx, 8               # перемещаем указатель
на следующий элемент массива чисел
        loop WokWork             # в цикле проходим по всем
элементам массива

ret

```

Выводы.

Была реализована программа организующая связь ассемблера с ЯВУ на примере программы построения частотного распределение попаданий псевдослучайных целых чисел в заданные интервалы.