ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
«РОССИЙСКИЙ УНИВЕРСИТЕТ ТРАНСПОРТА»  
(РУТ (МИИТ))

Институт транспортной техники и систем управления

Кафедра «Управление и защита информации»

ОТЧЁТ  
ПО ЛАБОРАТНОРНОЙ РАБОТЕ № 3

**«**Команды передачи управления**»**

По дисциплине «Машинно-ориентированные языки программирования»  
Вариант 5

**Выполнил: ст. гр. ТКИ – 341**

Боди Итшан

**Проверил: доц.**

Логинова Л. Н.

**Москва 2024**

**Цель работы**

Изучение команд передачи управления и получение навыка работы с ними.

**Постановка задачи**

* + 1. С помощью процедуры *Random* сформировать случайное целое число в диапазоне от 0 до 65535;
    2. Четное число (четное с точки зрения математики) записать в ЯП (ячейку памяти), начиная с нулевого элемента массива *data*. Нечетное число (нечетное с точки зрения математики) записать в ЯП, начиная с 128 элемента массива *data*. Все числа, которые больше или равны 50000 записать в ЯП, начиная с 256 элемента массива *data*. Все числа, которые меньше 10000 записать в ЯП, начиная с 384 элемента массива *data*;
    3. Повторять п.1-п.2, пока число распределенных данных по любому из признаков не превысит 127.

**Код программы**

#include <iostream>

#include <iomanip>

#include <random>

int main()

{

std::random\_device dev;

std::mt19937 generator(dev());

std::uniform\_int\_distribution<unsigned short> distribution(0, 65535);

const size\_t size = 512;

unsigned short mass[size];

unsigned short data[size] = {0};

for (size\_t i = 0; i < size; i++)

{

mass[i] = distribution(generator);

}

\_\_asm {

pushad

lea edi, mass

lea esi, data

mov ebx, 0

mov ecx, 0

mov edx, 0

dist:

mov ax, word ptr[edi]

cmp ax, 50000

jae big\_number

cmp ax, 10000

jb less\_number

test ax, 1

jz even\_number

jmp odd\_number

big\_number:

movzx ebx, ch

cmp ebx, 128

je end\_loop

mov [esi + ebx\*2 + 512], ax

inc ch

jmp increment\_number

less\_number:

movzx ebx, cl

cmp ebx, 128

je end\_loop

mov [esi + ebx\*2 + 768], ax

inc cl

jmp increment\_number

even\_number:

movzx ebx, dh

cmp ebx, 128

je end\_loop

mov [esi + ebx\*2], ax

inc dh

jmp increment\_number

odd\_number:

movzx ebx, dl

cmp ebx, 128

je end\_loop

mov [esi + ebx\*2 +256], ax

inc dl

jmp increment\_number

increment\_number:

add edi, 2

jmp dist

end\_loop:

popad

}

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

std::cout << std::setw(8) << mass[i] << " -----> " << std::setw(5) << data[i] << " i = " << i << std::endl;

}

return 0;

}

**Рисунки, иллюстрирующие работу кода**

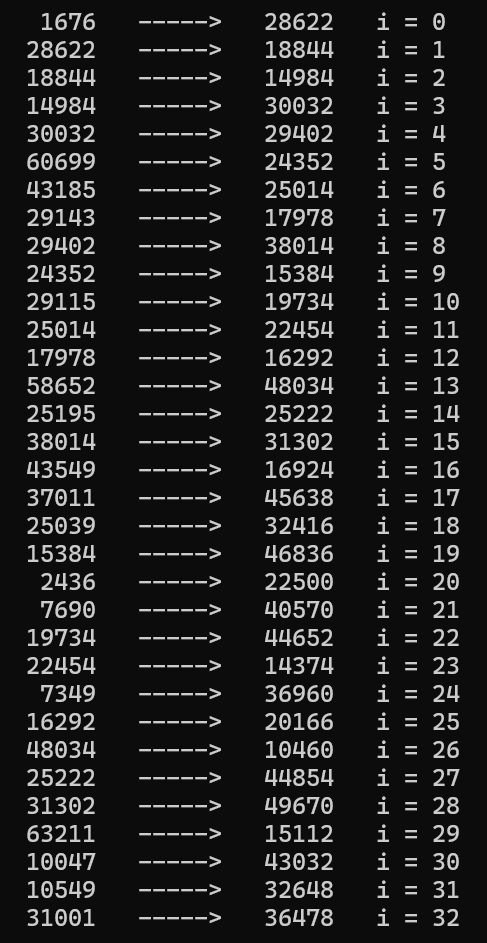


рис.1 - распределение элементов из одного массива в другой по заданному правилу с указанием индекса

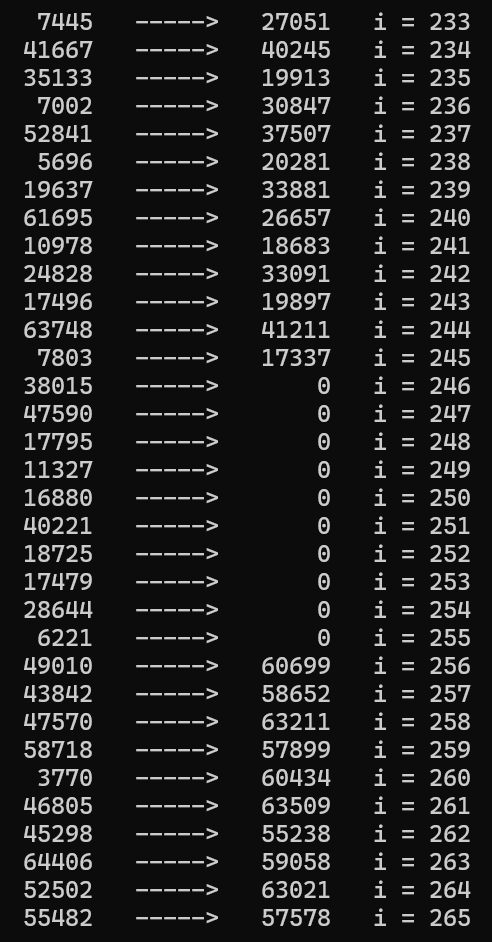


рис.2 - распределение элементов из одного массива в другой по заданному правилу на стыке индексов, согласно правилу

**Вывод**

В рамках данной лабораторной работы были изучены и освоены команды JMP (и её разновидности), LOOP (и её разновидности), TEST, а также работа с метками. Кроме того, было изучено использование массивов в языке ассемблера и принципы работы с данным типом данных.