Национальный исследовательский университет
Высшая школа экономики
Московский институт электроники и математики

Департамент прикладной математики
кафедра компьютерной безопасности

Отчет по лабораторной работе №4.1 по дисциплине "Языки ассемблера"

Вариант: 30

Выполнил:
Фролов О.В.

Москва, 2023

Оглавление

Домашнее задание 1.

Задание А4.1

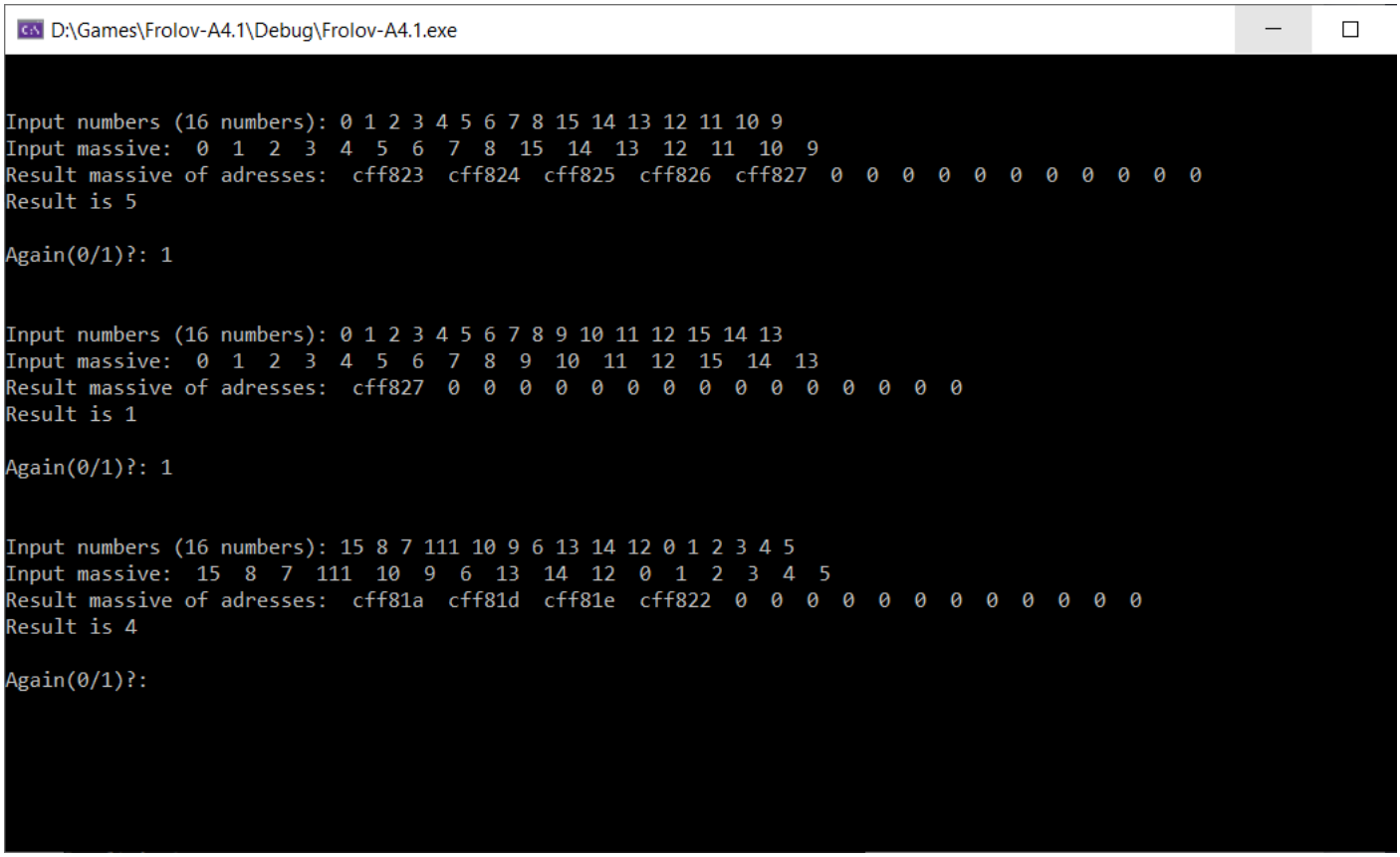
Дан массив A из 16 байтов. Если три последовательных элемента составляют невозрастающую последовательность, то скопировать средний элемент в байтовый массив B, а адрес (смещение) этого элемента — в массив C. Сосчитать количество таких элементов. (В последовательности 3,2,0,1,7,7,2,1,0 таких элементов четыре.)

```
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>
#include <string.h>
#include <locale.h>
#include <ctype.h>

int main() {
    setlocale(LC_ALL, "rus");

    int start = 1, flag=0;
    while (start == 1) {
        char a[16]= { 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0 }, b[14] = { 0,0,0,0,0,0,0,0,0,0,0,0,0,0 }; //14, а не 16, т.к. средних элементов может быть макс
        int c[16] = { 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0 }; //под адреса
        int res = 0;

        input:
        printf("\n\nInput numbers (16 numbers): ");
        scanf("%d %d %d %d %d %d %d %d %d %d %d %d %d %d %d %d", &a[0], &a[1], &a[2], &a[3], &a[4], &a[5], &a[6], &a[7], &a[8], &a[9], &a[10], &a[1
        getchar();

        __asm {
            lea esi, a //адрес начала массива a поместили в esi
            lea edi, b //адрес начала массива b поместили в edi
            lea edx, c //адрес начала массива c поместили в edx
            mov cx, 14 //макс. число повторений в цикле поместили в cx, 14 т.к. последние два элемента не образуют новых последовательностей из тре

            first:
                mov al, [esi] //элемент a[i] поместили в регистр al (как байт, поэтому +1 дальше идет для перехода к след. элементу)
                mov bl, [esi + 1] //элемент a[i+1] поместили в регистр bl(как байт)

                cmp al, bl //сравнили a[i] и a[i+1]
                jl second //переходим на second, если первый элемент < второго

                mov al, [esi+2] //элемент a[i+2] поместили в регистр al
                cmp bl,al //сравнили a[i+1] и a[i+2]
                jl second //переходим на second, если второй элемент < третьего

                //скопировать средний элемент (т.е. bl) в массив B, а адрес (смещение) этого элемента в C
                mov [edi], bl //поместили средний элемент a[i+1] в b[i]
                add esi, 2 //изменили адрес на адрес следующего элемента
                mov [edx], esi //поместили адрес (смещение) a[i+1] в c[i]
                sub esi, 2 //вернули прежний адрес
                inc edi //увеличили i для b[i]
```

```
43                  inc edi //увеличили i для b[i]
44                  add edx, 4 //увеличили i на 1 для c[i]
45                  add res, 1 //увеличили кол-во таких элементов
46              second:
47                  inc esi //увеличили i для a[i]
48                  dec cx //cx=cx-1
49                  cmp cx,0
50                  jne first //уйти на first, если cx не равно 0
51                  nop
52          }
53
54          printf("Input massive: ");
55          for (int i = 0; i < 16; ++i) {
56              printf(" %d ", a[i]);
57          }
58          printf("\nResult massive of adresses: ");
59          for (int i = 0; i < 16; ++i) {
60              printf(" %x ", c[i]);
61          }
62          printf("\nResult is %d", res);
63          printf("\n\nAgain(0/1)?: ");
64          if (scanf("%d", &start)) {
65              getchar();
66              if (start == 1)
67                  start = 1;
68              else
69                  start = 0;
70          }
71      }
72      exit(0);
73  }
```

Принцип работы: Сначала пользователь вводит 16 чисел.
Далее выполняется ассемблерная вставка: в регистры esi, edi, edx помещаются адреса, с которых начинаются массивы a,b,c соответственно. В cx помещается максимальное кол-во итераций последующих процедур (14, т.к. чисел всего 16, а проверяются тройками). Далее в al помещается первое число в текущей итерации (в зависимости от адреса, лежащего в esi), а в bl второе аналогично, они сравниваются, если первое оказывается меньше второго, то происходит переход на second, где увеличивается счетчик итераций, лежащий в cx, и если он не равен 0, то происходит переход на first, но перед этим увеличивается адрес esi для перехода к следующей тройке элементов. Если первое число оказалось больше или равно второму, то в al помещается третье число, а дальше сравнивается bl и al, если второе число окажется меньше третьего, то произойдет переход на second, иначе средний элемент будет помещен в массив b, а адрес (смещение) этого элемента в массив c, счетчик этих элементов res будет увеличен на 1, а дальше переход на second.
После всех процедур пользователю вернется количество искомых элементов, а далее поступает вопрос о том, нужно ли повторить работу программы, или же можно завершить выполнение программы.

1. Тесты


D:\Games\Frolov-A4.1\Debug\Frolov-A4.1.exe
Input numbers (16 numbers): 0 1 2 3 4 5 6 7 8 15 14 13 12 11 10 9
Input massive: 0 1 2 3 4 5 6 7 8 15 14 13 12 11 10 9
Result massive of adresses: cff823 cff824 cff825 cff826 cff827 0 0 0 0 0 0 0 0 0 0 0
Result is 5
Again(0/1)?: 1
Input numbers (16 numbers): 0 1 2 3 4 5 6 7 8 9 10 11 12 15 14 13
Input massive: 0 1 2 3 4 5 6 7 8 9 10 11 12 15 14 13
Result massive of adresses: cff827 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
Result is 1
Again(0/1)?: 1
Input numbers (16 numbers): 15 8 7 111 10 9 6 13 14 12 0 1 2 3 4 5
Input massive: 15 8 7 111 10 9 6 13 14 12 0 1 2 3 4 5
Result massive of adresses: cff81a cff81d cff81e cff822 0 0 0 0 0 0 0 0 0 0 0 0
Result is 4
Again(0/1)?: