**Лабораторная работа №2**

Цель работы: На языке assembler научиться работать с массивами, производить различные операции с их элементами, упаковывать массив и обеспечивать доступ к его элементам, сортировать массивы

**Задание 1**

Вариант 9: Условие: посчитать количество положительных элементов массива; Данные: 3, -14, 32, 16, -52, -8, 7, 26; Ответ: 5

Переберем все элементы массива и сравним каждый с нулем. Если элемент больше нуля, то увеличиваем счетчик

Пусть индекс элементов хранится в регистре EBX, а счетчик положительных элементов в регистре DL, значение которого выводится при выполнении программы. Запишем программу:

%include "io.inc"

; посчитать количество положительных элементов в массиве

;ответ: 5

section .text

global CMAIN

CMAIN:

LEA EBX, [A] ;загружаем индекс первого элемента массива

MOV ECX, 8 ;задаем счетчику цикла значение количества элементов массива

XOR DL, DL ;обнуляем счетчик положительных элементов

next:

MOV AL, [EBX] ;Загружаем значение элемента массива

CMP AL, 0 ;сравниваем его с нулем

JNG skip ;если он меньше или равен нулю, то пропускаем

INC DL ;Если он больше, то увеличиваем счетчик

skip:

INC EBX ;увеличиваем индекс, чтобы рассмотреть следующие значения

LOOP next ;вводим цикл для перебора всех элементов массива

PRINT\_UDEC 2, DL ;выводим количество элементов положительных элементов массива

ret

section .data

A: DB 3, -14, 32, 16, -52, -8, 7, 26

После выполнения программа выдает ответ 5, что соответствует условию задачи

**Задание 2**

Вариант 2: Задание: Написать программу, реализующую доступ к упакованному массиву с разрядностью элементов 5 бит. Данные: 2, 14, 23, 4, 7, 18, 1, 30

Переведем все числа в двоичную систему и оставим только младшие 5 бит. Запишем все числа подряд в одну линию и разобьем на три части по 8 бит. Каждые 8 бит переведем, как отдельное число в шестнадцатеричную систему и запишем в массив. Для того, чтобы вытащить нужное значение будем выбирать нужные биты из разных элементов с помощью маски, менять их положение с помощью операций сдвига в зависимости от того, старшие они или младшие.

Запишем программу, которая выводит исходные элементы массива:

%include "io.inc"

;доступ к упакованному массиву

;2, 14, 23, 4, 7, 18, 1, 30

section .text

global CMAIN

CMAIN:

LEA ESI, [A]

;Для извлечения первого элемента

MOV AL, [ESI]

MOV DL, 0xF8 ;маска

AND AL, DL ;маскируем

SHR AL, 3 ;сдвигаем биты вправо для получение искомого числа

PRINT\_UDEC 1, AL ;выводим искомое число

NEWLINE

;Для извлечения второго элемента

;достаем старшие 3 бита

MOV AL, [ESI]

MOV DL, 0x07

AND AL, DL

SHL AL, 2 ;ставим извлеченнные биты на нужные позиции

MOV BL, AL ;сохряняем старшие три бита

;достаем младшие 2 бита из второго элемента

MOV AL, [ESI + 1]

MOV DL, 0xC0

AND AL, DL

SHR AL, 6

OR AL, BL ;накладываем старшие и младшие биты, чтобы получить число полностью

PRINT\_UDEC 1, AL

NEWLINE

;Для извлечения третьего элемента

MOV AL, [ESI + 1]

MOV DL, 0x3E

AND AL, DL

SHR AL, 1

PRINT\_UDEC 1, AL

NEWLINE

;Для извлечения четвертого элемента

;достаем 1 старший бит

MOV AL, [ESI + 1]

MOV DL, 0x01

AND AL, DL

SHL AL, 4

MOV BL, AL

;достаем 4 младших бита

MOV AL, [ESI + 2]

MOV DL, 0xF0

AND AL, DL

SHR AL, 4

OR AL, BL

PRINT\_UDEC 1, AL

NEWLINE

;Для извлечения пятого элемента

;достаем старшие 4 бита

MOV AL, [ESI + 2]

MOV DL, 0x0F

AND AL, DL

SHL AL, 1

MOV BL, AL

;достаем 1 младший бит

MOV AL, [ESI + 3]

MOV DL, 0x80

AND AL, DL

SHR AL, 7

OR AL, BL

PRINT\_UDEC 1, AL

NEWLINE

;Для извлечения шестого элемента

MOV AL, [ESI + 3]

MOV DL, 0x7C

AND AL, DL

SHR AL, 2

PRINT\_UDEC 1, AL

NEWLINE

;Для извлечения седьмого элемента

;достаем старшие 2 бита

MOV AL, [ESI + 3]

MOV DL, 0x03

AND AL, DL

SHL AL, 3

MOV BL, AL

;достаем 3 младших бита

MOV AL, [ESI + 4]

MOV DL, 0xE0

AND AL, DL

SHR AL, 5

OR AL, BL

PRINT\_UDEC 1, AL

NEWLINE

;Для извлечения восьмого элемента

MOV AL, [ESI + 4]

MOV DL, 0x1F

AND AL, DL

PRINT\_UDEC 1, AL

NEWLINE

ret

A: DB 0x13, 0xAE, 0x43, 0xC8, 0x3E

После выполнения программа выдает элементы исходного массива

**Задание 3**

Вариант 1. Задание: Написать программу, реализующую алгоритм сортировки выбором. Данные: 7, 4, 9, 2, 0, 3, 1, 6, 8, 5. Ответ: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9.

Поочередно сохраняем индекс позиций в порядке возрастания и ищем в массиве элемент, подходящий на это место от наименьшего к наибольшему.

Запишем программу, которая в случае верного выполнения выведет отсортированный массив:

%include "io.inc"

section .text

global CMAIN

CMAIN:

LEA EBX, [A] ; загружаем индекс первого элемента массива

MOV ESI, EBX ; сохраняем первый индекс для поочередного прохода

MOV EDI, EBX ; сохраняем индекс для вывода массива

MOV DL, 0 ; присваиваем самый маленький элемент массива

MOV CX, 10 ; задаем счетчику значение количества элементов массива

next:

MOV AL, [EBX] ; вытаскиваем значение n-го элемента

INC EBX ; увеличиваем индекс, чтобы потом работать со следующими элементами

CMP AL, DL ;сравниваем n-ый элемент с наименьшим

JNZ next ;если не совпадает, и рассматриваем следующий элемент, если совпадает, то делаем замену

MOV AH, [ESI] ;сохраняем n-ый элемент

MOV [ESI], AL ;записываем на его место наименьший

SHR AX, 8 ;сдвигаем значение регистра для получения n-го элемента

MOV [EBX -1], AL ; записываем n-ый элемент на бывшее место наименьшего

INC ESI ;сужаем круг неотсортированного массива

MOV EBX, ESI ;задаем индекс первого элемента неотсортированного массива

INC DL ;увеличиваем значение самого маленького элемента

DEC CX ;уменьшаем счетчик

CMP CX, 0 ;если 0, то завершаем сортировку

JA next ;если нет, то переходим к следующему шагу

;выведем отсортированный массив

MOV EBX, EDI

MOV ECX, 10

XOR ESI, ESI

prar:

MOV AL, [EBX + ESI]

PRINT\_DEC 2, AL

INC ESI

LOOP prar

ret

section .data

A: DB 7, 4, 9, 2, 0, 3, 1, 6, 8, 5

После выполнения программа выдает упорядоченный по возрастанию массив

Вывод: я научился работать с массивами, производить различные операции с их элементами, упаковывать массив и обеспечивать доступ к его элементам, сортировать массивы на языке assembler