Лабораторная работа № 6

Создание макросов для ввода и вывод данных

Цель работы: Практическое овладение навыками разработки программного кода на языке Ассемблер. Изучение приемов разработки макроопределений.

Порядок выполнения работы

1. Создать рабочую папку для текстов программ на ассемблере и записать в нее файлы tasm.exe, tlink.exe, rtm.exe и td.exe из пакета tasm, а также файл с исходным текстом программы на ассемблере, который сохранить с именем prog6.asm.
2. Создать загрузочный модуль, загрузить его в отладчик и выполнить программу в пошаговом режиме.

Содержание отчета:

1. Цель работы.
2. Постановка задачи.
3. Теоретическая часть – словесный алгоритм макроса ввода-вывода.
4. Листинг программы.
5. Пояснения к программе.
6. Результат работы программы.
7. Вывод.

Постановка задачи

1. **Создать макросы для ввода и вывода чисел (двух, трех и четырехзначных).**
2. **С использованием макросов выполнить задание, соответствующее варианту.**
3. **Исходные данные вводятся с клавиатуры (n, c, d).**
4. **Результаты выводятся на экран.**

Варианты

Вариант 1

Ввести с клавиатуры последовательность из N чисел, размером в слово. Значение N задается с клавиатуры, но должно быть не больше 10. Найти, сколько элементов введенной последовательности удовлетворяет условию: c<=a[i]<=d. Значение c, d задается с клавиатуры. Результат вывести на экран.

Вариант 2

Ввести с клавиатуры последовательность из N чисел, размером в слово. Значение N задается с клавиатуры, но должно быть не больше 10. Найти сумму квадратов всех положительных элементов введенной последовательности удовлетворяющих условию: a[i]>=c*d. Значение c, d задается с клавиатуры. Результат вывести на экран.

Вариант 3

Ввести с клавиатуры последовательность из N чисел, размером в слово. Значение N задается с клавиатуры, но должно быть не больше 10. Найти, сколько положительных, отрицательных и нулевых элементов введенной последовательности удовлетворяет условию: c<=a[i]<=d.
Значение c, d задается с клавиатуры. Результат вывести на экран.

Вариант 4
Ввести с клавиатуры последовательность из N чисел, размером в слово. Найти произведение последних L отрицательных элементов введенной последовательности, удовлетворяющих условию: c<=a[i]<=d. Значение n, l, c, d задается с клавиатуры. Результат вывести на экран.

Вариант 5
Ввести с клавиатуры последовательность из N чисел размером в байт. N задается с клавиатуры, но должно быть не больше 15. Найти в массиве байт, являющийся 4-м нечетным байтом. Вывести на экран массив начиная с этого элемента.

Вариант 6
Ввести с клавиатуры последовательность из N чисел, размером в слово. Найти, сколько положительных элементов введенной последовательности удовлетворяет условию: a[i]>=c/d. Значение n, c, d задается с клавиатуры. Результат вывести на экран.

Вариант 7
Ввести с клавиатуры последовательность из N чисел, размером в слово. Найти, сколько отрицательных элементов введенной последовательности удовлетворяет условию:a[i]>=c+d. Значение n, c, d задается с клавиатуры. Результат вывести на экран.

Вариант 8
Ввести с клавиатуры последовательность из N чисел, размером в слово. Найти сумму пяти отрицательных элементов последовательности, без минимального элемента. Значение n, задается с клавиатуры. Вывести на экран получившееся значение.

Вариант 9
Ввести с клавиатуры последовательность из N чисел, размером в слово. Найти сумму трех положительных элементов последовательности, без максимального элемента. Значение n, задается с клавиатуры. Вывести на экран получившееся значение.

Вариант 10
Ввести с клавиатуры последовательность из N чисел размером в байт. N задается с клавиатуры, но должно быть не больше 15. Найти в массиве 3-й по порядку нулевой байт. Вывести на экран массив начиная с этого элемента.

Вариант 11
Ввести с клавиатуры последовательность из N символов размером в байт. N задается с клавиатуры, но должно быть не больше 15. Найти в массиве 4-й по порядку байт из числа тех, которые ниже 20h. Вывести на экран массив начиная с этого элемента.

Вариант 12
Ввести с клавиатуры последовательность из N символов размером в байт. N задается с клавиатуры, но должно быть не больше 15. Найти в массиве 3-й по порядку байт код символа ‘$’(24h). Вывести на экран массив начиная с этого элемента.

Вариант 13
Ввести с клавиатуры последовательность из N символов размером в байт. N задается с клавиатуры, но должно быть не больше 15. Найти в массиве байт, следующий за 3-м кодом символа ‘;’(3Bh). Вывести на экран массив начиная с этого элемента.

Вариант 14
Ввести с клавиатуры последовательность из N чисел размером в байт. N задается с клавиатуры, но должно быть не больше 15. Найти в массиве байт, следующий за 3-м отрицательным байтом. Вывести на экран массив начиная с этого элемента.

Вариант 15
Ввести с клавиатуры последовательность из N чисел, размером в слово. Найти сумму первых K элементов введенной последовательности удовлетворяющих условию: c<=a[i]<=d. Значение k, c, d задается с клавиатуры. Результат вывести на экран.

Вариант 16
Ввести с клавиатуры последовательность из N символов размером в байт. N задается с клавиатуры, но должно быть не больше 15. Найти в массиве байт, следующий за 3-м кодом пробела (20h). Вывести на экран массив начиная с этого элемента.

Вариант 17
Ввести с клавиатуры последовательность из N символов размером в байт. N задается с клавиатуры, но должно быть не больше 15. Найти в массиве 3-й байт из числа тех, кто выше (10h). Вывести на экран массив начиная с этого элемента.

Вариант 18
Ввести с клавиатуры последовательность из N чисел, размером в слово. Значение N задается с клавиатуры, но должно быть не больше 10. Найти сумму квадратов всех отрицательных элементов последовательности. Результат вывести на экран.

Вариант 19
Ввести с клавиатуры последовательность из N чисел размером в слово. N задается с клавиатуры, но должно быть не больше 10. Найти максимальное и минимальное из введенных чисел, значение которых вывести на экран.

Вариант 20
Ввести с клавиатуры последовательность из N чисел, размером в слово. Значение N задается с клавиатуры, но должно быть не больше 10. Найти, сколько элементов введенной последовательности удовлетворяет условию: a[i]>=d. Значение d задается с клавиатуры. Результат вывести на экран.

Теоретическая часть

Макроопределение

Макроопределение **(macro),** это символическое имя, присвоенное одному или нескольким утверждениям языка ассемблер. Однажды определенное, оно может вызываться сколько угодно раз в программе. Когда происходит вызов макроопределения, копия утверждений, определенных данным макроопределением, вставляется прямо в программу.

Макроопределения обычно выполняются быстрее чем процедура, имеющая те же самые команды. Это происходит потому, что для вызова процедуры необходимы дополнительные команды CALL и RET, что заставляет процессор организовывать переход на другую ветвь и делать возврат из нее в исходную точку. Но их использование увеличивает объем программного кода, так как каждый вызов макроопределения вставляет в программу код самого макроопределения. Макроопределения может быть написано в любом месте программы с помощью директив **macro** и **endm.**

Синтаксис

Имя **macro parameter_1, parameter_2…** **;;** **начало макроопределения**
Список команд
endm **;;** **конец макроопределения**

Именем может быть любой идентификатор. Для лучшего выделения макроопределения в его имени добавляют префикс (строчная буква ‘m’) - **m**Puthchar.

Параметры **parameter_1, parameter_2**… - или фиктивные параметры, определяют место для реальных значений, которые заполняются при вызове макроопределения.

Параметры не являются обязательными, их может быть любое количество, которое можно разместить на одной строке, отделив друг от друга запятыми.

Все команды до директивы **endm** считаются частью макроопределения. Эти команды не ассемблируются, пока макроопределение действительно не будет вызвано.

Для выделения комментариев, в макроопределениях два раза ставиться точка с запятой (;;). При этом комментарии появляются только в написанном макроопределении, а не в тех фрагментах, которые вставляются в тело программы.

Пример. Макрос отображения на экране символа

```
mPuthchar macro     начало макроопределения
mov ah,2
int 21h
endm                ;; конец макроопределения
```

Макроопределение вызывается при размещении его имени в исходном коде программы, возможно, с необходимыми аргументами.

Синтаксис:

Имя **parameter_1, parameter_2…**

Использование макроопределения **mPuthchar**

Исходный код	**Расширенный код**
Mov dl,’A’	Mov dl,’A’
mPuthchar	***mov ah,2 ;;команды тела макроопределения***
	int 21h
Mov dl,’*’	Mov dl,’*’
mPuthchar	***mov ah,2;;команды тела макроопределения***
	int 21h

Каждый аргумент является значением, передаваемым в макроопределение. Это значение заменяет соответствующие параметры при вызове. Порядок аргументов должен соответствовать порядку параметров, но число аргументов необязательно равно числу этих параметров. Если аргументов слишком много, то Assembler выдаст предупреждение, а если аргументов мало, оставшиеся параметры будут пропущены.

Пример. Макроопределение для записи строки на экран

Здесь требуется один параметр, называемый **string** и ссылающийся на строку, которую необходимо записать на экране. Параметр **string** замещается при каждом вызове

```
Msg1 db “This is massage 1.”, 0Dh,0Ah,’$’
Msg2 db “This is massage 2.”, 0Dh,0Ah,’$’
Msg3 db “This is massage 3.”, 0Dh,0Ah,’$’
…
mDisplayStr macro string
        push ax
        push dx
        mov ah,9
        mov dx, offset string
        int 21h
        pop dx
        pop ax
endm
…
mDisplayStr Msg1
mDisplayStr Msg2
mDisplayStr Msg3
```

Параметр **string** замещается при каждом вызове макроопределения. При отображении трех строк вызывается три раза, используя каждый раз различные аргументы.

Аргументы макроопределения могут быть: **непосредственными значениями, операндами памяти или регистрами размером 8 битов**, делая удобным вызов процедуры.

Пример. Макроопределение размещает курсор в указанной строке и столбце экрана

```
mGotoRowCol 10,20          ;; непосредственные значения
mGotoRowCol row, column    ;; операнды памяти
mGotoRowCol ch,cl          ;; регистры

mGotoRowCol macro row: REQ, column: REQ
        push ax
        push bx
        push dx
        mov bx,0
        mov ah,2
        mov dh, row
        mov dh, column
        int 10h
        pop dx
        pop bx
        pop ax
endm
```

REQ – спецификатор, который устанавливает необходимые параметры.

В нашем примере при установке курсора обязательными параметрами являются – номер строки и номер столбца, без них выполнение макроса невозможно.

Если макроопределение вызывается без необходимых параметров, ассемблер генерирует ошибку.

Вложенные макроопределения

В языке ассемблер предусмотрена компоновка существующих макроопределений

Макроопределение, включенное в другое называется **вложенным макроопределением.** Файл листинга автоматически показывает уровни вложенности для всех утверждений макроопределения.

Пример Макроопределение отображает строки в заданном месте экрана

```
mDisplayRowCol macro row, column, string
mGotoRowCol row, column ;; вызовmGotoRowCol
mDisplayStr string              ;; отображает строки
endm
…
.data
greeting db “Hello from row 10, column 15. $’’
.code
mDisplayRowCol 10, 15, greeting
…
```

Мы вызываем макрос **mDisplayRowCol** с параметрами строки, равной 10, столбца, равного 15 и строкой с именем greeting. Уровни вложенности для всех утверждений 2, что говорит о том, что макроопределение вызвано из другого макроопределения.

В сегменте кода организация вызова макроопределения происходит при помощи размещения имени макроопределения в исходной программе. В каждой точке программ, где появляется имя макроопределения, вставляются команды из тела макроса. Вставка кодов происходит при первом прохождении ассемблером исходного файла, и они будут присутствовать в листинге, сгенерированном ассемблером.