21. Макроопределения.

Макроопределение (макрос) - именованный участок программы, который ассемблируется каждый раз, когда его имя встречается в тексте программы.

Определение:

```
имя MACRO параметры
...
ENDM
```

Директива присваивания `=`.

Директива присваивания служит для создания целочисленной макропеременной или изменения её значения и имеет формат:

```
Макроимя = Макровыражение
```

- **Макровыражение (или Константное выражение)** - **выражение, вычисляемое препроцессором**, которое может включать целочисленные константы, макроимена, вызовы макрофункций, знаки операций и круглые скобки, **результатом вычисления которого является целое число**
- Операции: арифметические (+, -, *, /. MOD), логические, сдвигов, отношения (сравнения)

Пример:

```
K=10
A DW K ;эквивалентно A DW 10
K=K+4
B DW K ;эквивалентно B DW 14
```

Директивы отождествления `EQU`, `TEXTEQU`.

Директива для представления текста и чисел:

```
Макроимя EQU нечисловой текст и не макроимя ЛИБО число
Макроимя EQU <Операнд>
```

`TEXTEQU` никак не обрабатывает выражение и просто присваивает его как текстовую строку левой части выражения (макроимени).

```
Макроимя TEXTEQU Операнд
```

Пример:

```
X EQU [EBP+8]
MOV ESI,X
```

Макрооперации.

Помимо очевидных арифметических операций, есть специфические операции

- `%` - вычисление выражения перед представлением числа в символьной форме (позволяет подставлять куда-либо результаты вычислений в качестве текстового значения)
- `<>` - подстановка текста без изменений (без конкатенаций или арифметических вычислений, т. е. не изменяя операнд и не интерпретируя его как выражение)
- `&` - склейка текста (конкатенация слов)
- `!` - считать следующий символ текстом, а не знаком операции (можно ставить перед знаками `<`, `>`, `=` и т.д.)
- `;;` - исключение строки из макроса (**комментарий в макросе**). Если в макросах ставить `;`, которые мы обычно используем для строковых комментариев, то эта строка попадёт в результат работы макроса!

Блоки повторения.

- Повтор фиксированное число раз: `REPT число … ENDM`
- Подстановка фактических параметров по списку на место формального: `IRP` или `FOR` (в разных версиях MASM'a): `IRP form,<fact_1[,fact_2,...]> … ENDM` - перебираем значения, которые по очереди будут подставляться вместо формального имени (которое мы должны указать первым операндом) и это формальное имя можно использовать в теле цикла кол-во итераций цикла = кол-во параметров в угловых скобках.
  - **Пример**

```
; Пример. Поместить в стек регистры AX, BX, CX, DX.
IRP reg, <ax, bx, cx, dx>
push reg
ENDM
; Будут сгенерированы команды:
; push ax
; push bx
; push cx
; push dx
```

- Подстановка символов строки на место формального параметра: `IRPC` или `FORC` (в разных версиях MASM'a): `IRPC form,fact … ENDM` - поочерёдно перебираем все символы (элементы) строки fact.
- Цикл с условием: `WHILE cond … ENDM`:

  Условие cond должно возвращать значение типа bool пока cond истинно, выполняется подстановка собственного содержимого в текст программы

Директивы условного ассемблирования.

Похожи на условные операторы из языков высокого уровня.

Директивы:

- `IF`:

```
IF c1
...
ELSEIF c2    ;; может отсутствовать, как и в условных операторах языков высокого уровня
...
ELSE         ;; может отсутствовать, как и в условных операторах языков высокого уровня
...
ENDIF
```

- `IFB <par>` - истинно, если параметр не определён
- `IFNB <par>` - истинно, если параметр определён
- `IFIDN <s1>,<s2>` (IF IDeNtical) - истинно, если строки совпадают
- `IFDIF <s1>,<s2>` - истинно, если строки разные
- `IFDEF/IFNDEF <name>` - истинно, если имя объявлено/не объявлен