Data types

- Byte -8bits
- Word -16bits
- Double Word -32bits
- Quad Word -64 bits
- Double Quad Word -128bits
- NASM is Case Sensitive Synatx
- `mov rax,message` (move address to rax)
- `mov rax, [message]`(move value to rax)

Defining Initialized Data in NASM

```
db    0x55                ; just the byte 0x55
db    0x55,0x56,0x57      ; three bytes in succession
db    'a',0x55            ; character constants are OK
db    'hello',13,10,'$'   ; so are string constants
dw    0x1234              ; 0x34 0x12
dw    'a'                 ; 0x61 0x00 (it's just a number)
dw    'ab'                ; 0x61 0x62 (character constant)
dw    'abc'               ; 0x61 0x62 0x63 0x00 (string)
dd    0x12345678          ; 0x78 0x56 0x34 0x12
dd    1.234567e20         ; floating-point constant
dq    0x123456789abcdef0  ; eight byte constant
dq    1.234567e20         ; double-precision float
dt    1.234567e20         ; extended-precision float
```

Declare Uninitialized Data

```
buffer:         resb    64              ; reserve 64 bytes
wordvar:        resw    1               ; reserve a word
```

Special Tokens

- $ - evaluates to the current line

- $$ - evaluates to the beginning of current section

```
message         db      'hello, world'
msglen          equ     $-message
```

Data:

```
zerobuf:        times 64 db 0
```

Instruction:

```
times 100 movsb
```

- Times :

  Two other common methods used for declar-ing arrays of data are the TIMES directive and the use of string literals. The TIMES directive tells the assembler to duplicate an expression a given number of times. For example, the statement “TIMES 4 DB 2” is equivalent to “2, 2, 2, 2”.

  ```
  times 8 db 0x00
  ```