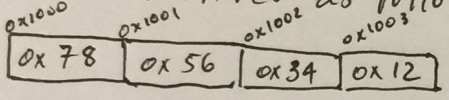
CS271 WK 8 Notes

Data-Related Operators

- OFFSET: returns distance in bytes from the top of the memory / var declaration segment
- PTR operator: provides the ability to move parts of a larger variable.
  - → use WORD PTR <doubleword_x>
    - ↳ allows access to the higher ("last") 2 bytes of the 4-byte variable.
    - ↳ in ex. above, if doubleword = 12345678h and then the result would be 5678h as bytes stored flipped. (Little endian)
  - → if data is stored as follows



this is the same as the DWORD 0x12345678 in LE.

then word (2-bytes) starting at [data+1] would be 0x34 0x56 because it "corrects" endianness.

- ~~SIZE~~ TYPE returns the size of an item of that data type.
  - ↳ usage ~~SIZE~~ TYPE <var_name>
- $IZEOf returns number of bytes that it takes to rep data in mem.
  - SIZEOF = type · number
- LENGTHOf returns number of elements

Handy List Trick:

- mov <reg>, <list_name>[esi * TYPE list_name]

Strings in ASM

- lodsb: Load string byte
  stosb: store string byte

→ cld: clear dir. flg
std: set dir flg

- lodsb: moves byte at [esi] into AL, increments (DF=0) or decrements (DF=1) ESI.
- stosb: stores byte at ~~ESI~~ in AL into [ESI], changes flag.

Lower Level Programming

- All input treated as strings of characters.

Reverse Polish Notation

- → a + (b - c) * (d + e) (infix notation)
- → a bc- de+ * + (RPN)
  - ↳ abc-de+*+
- operands in same order
- operators may be in diff order

FPU and RPN

- FPU for IA-32 is 0-address:
- a - b * c → abc*-

```
push a
push b
push c
mul
sub
```

operands are top two on stack

IA-32 FPU

- 0-address - machine
- converts WORD and DWORD to REAL10
- uses a push-down stack
- uses registers ST0..ST7 each of type 80-bit
- ST means top of stack

push down deletes last item if pushed too far

- First instruction MUST be FInit
- FLD = Push to FP reg
- FST = store to mem, DO not pop
- FSTP = store to mem, pop