Simple commands of assembly language to understand how assembly language works by commands

1. Data Movement:
   - `mov`: Moves data from one location to another.
   - `push`: Pushes data onto the stack.
   - pop: Pops data off the stack into a register or memory location.
   - `lea`: Loads the effective address of a memory location into a register.
2. Arithmetic and Logical Operations:
   - `add`: Adds two values.
   - `sub`: Subtracts one value from another.
   - `mul`: Multiplies two values.
   - `div`: Divides one value by another.
   - and: Performs a bitwise AND operation.
   - `or`: Performs a bitwise OR operation.
   - `xor`: Performs a bitwise XOR operation.
   - not: Performs a bitwise NOT operation.
3. Conditional Branching and Jumps:
   - `cmp`: Compares two values and sets the flags accordingly.
   - `je`, `jne`: Jump if equal, jump if not equal.
   - `jg`, `jge`, `jl`, `jle`: Jump if greater, jump if greater or equal, jump if less, jump if less or equal.
   - `jmp`: Unconditional jump to a specified label or memory address.
4. Looping:
   - `loop`: Decrements the loop counter and jumps to a specified label until the counter reaches zero.
   - `inc`: Increments the value of a register or memory location.
   - `dec`: Decrements the value of a register or memory location.
5. Procedure Calls:
   - `call`: Calls a subroutine or function.
   - `ret`: Returns from a subroutine.
6. Input and Output:

- `in`: Reads a byte or word from an input port.
- `out`: Writes a byte or word to an output port.

7. String Manipulation:
    - `movsb`: Moves a byte from the source string to the destination string.
    - `stosb`: Stores a byte in the destination string.
    - `cmpsb`: Compares a byte in the source string with a byte in the destination string.