Intro to Assembler Programming

- **Assembler**
  - **Assembler**:
    - **Assembler** = a **computer program** that translates ***mnemonic* codes** into **computer instructions** (stored in **binary representation** !!)
  - A **mnemonic** helps one remember a complicated fact

    Example:

    - To **remember** the **names** of the **5 great lakes**, you can use the mnemonic **HOMES**:
      - H = Huron
      - O = Ontario
      - M = Michigan
      - E = Erie
      - S = Superior

  - **Assembler instructions** are written using **mnemonics**

    Example assembler instructions:

```
add.b D0, D1      mnemonic for: add the byte operands in D0
                                and D1 and store result in D1
```

- **Format of an assembler program**
  - What an **assembler program** look like:

```
[label1] line1  [comment1]
[label2] line2  [comment2]
[label3] line3  [comment3]
.... and so on
end
```

The bracket `[ ... ]` denotes an ***optional* field**

- A **line** in the **assembler program** can contain:

  - ***one* assembler *directive*,** or
  - ***one* assembler *instruction*** (= **mnemonic** for a computer instruction)

- **Assembler Directives**

  - **Assembler *directive*:**

    - **Assembler *directive*** = a **command** given to the **assembler program**

      - An **assembler *directive*** tells (= directs) the **assembler** to ***do* something**

  - **Sample directives:**

    - Tell assembler to **start** a program at a certain location in memory
    - Tell assembler to **define** a symbolic constant
    - Tell assembler to **reserve space** for variables
    - Tell assembler to **stop** assembling