**The Assembler**

* Assembly code (jr $31) → assembler → machine code (01101…)
* Any translation involves 2 phases
  + Analysis – understand what is meant by the source string
  + Synthesis – output equivalent target string
* **Assembly file** – stream of characters
  + First group the chars into meaningful tokens
    - E.g. label, hex #, .word, etc.
  + Group tokens into instructions (if possible); aka. Analysis
  + If the tokens do not form sensible instructions, output ‘ERROR’ to stderr
    - Focus on checking correct cases rather than error cases
* How to assemble program branches?
  + Assemble in 2 passes
  + Pass 1:
  + Group tokens into instructions
  + Record addresses of labelled instructions in a “symbol table” – list of (label, address) pairs
    - A line of assembly can have more than one label
    - A word can be labelled after the end of a file
  + Pass 2:
  + Translate each instruction into machine code
  + If an instruction refers to a label, look up the corresponding address in symbol table
  + Output assembled MIPS code to stdout
  + Output symbol table to stderr