# Assembler – Iteration 1

I started writing my code paying close attention to the Decomposing the Problem section. For this reason, the initial code I typed looked like this:

def main(asmfile, out\_format):

with open(asmfile, "rt") as file:

text = file.read()

# Now the text is available

# 1. PERFORM TEXT NORMALISATION

normalised\_text = normalise\_text(text)

print(normalised\_text)

# 2. SPLIT DOCUMENT INTO SECTIONS

section\_dict = split\_into\_sections(normalised\_text)

pprint(section\_dict)

# 3. DIVIDE LINES AND CONTEXTUALISE

config\_dict, instruction\_list = divide\_and\_contextualise(section\_dict)

# 4. RECORD LABELS/VARIABLES

mem\_table = record\_labels\_and\_variables(instruction\_list)

# 5. CONVERT EACH LINE TO BYTES

place\_memory\_addresses(mem\_table, instruction\_list)

bytecode = b""

bytecode += encode\_metadata(config\_dict)

bytecode += encode\_instruction\_list(instruction\_list, mem\_table)

This is an outline of the main structure of the program. At present, none of those functions exist, and the purpose of initially writing this was simply to start with an overall view of the program. In order to permit testing, as I was moving through the program I commented out lines calling functions that did not yet exist, and adding in temporary print statements to view the output of the functions.