

ASSEMBLER IN PYTHON:

```
import os
import copy

path_dir = os.path.dirname(os.path.abspath(__file__)) # directory of the
folder from Python script
#####
#####
name_file = 'enter file name here'
#####
#####
asm_path = os.path.join(path_dir, f"{name_file}.asm")
bin_path = os.path.join(path_dir, f"{name_file}.hack")

# Symbolic: dest = comp; jump
# Binary:
# acccccc
comp_dict = {
    '0': '0101010', '1': '0111111', '-1': '0111010', 'D': '0001100',
    'A': '0110000', 'M': '1110000', '!D': '0001101', '!A': '0110001',
    '!M': '1110001', '-D': '0001111', '-A': '0110011', '-M': '1110011',
    'D+1': '0011111', 'A+1': '0110111', 'M+1': '1110111', 'D-1': '0001110',
    'A-1': '0110010', 'M-1': '1110010', 'D+A': '0000010', 'D+M': '1000010',
    'D-A': '0010011', 'D-M': '1010011', 'A-D': '0000111', 'M-D': '1000111',
    'D&A': '0000000', 'D&M': '1000000', 'D|A': '0010101', 'D|M': '1010101'
}

# ddd
dest_dict = {
    'null': '000', 'M': '001', 'D': '010', 'MD': '011',
    'A': '100', 'AM': '101', 'AD': '110', 'AMD': '111'
}

# jjj
jump_dict = {
    'null': '000', 'JGT': '001', 'JEQ': '010', 'JGE': '011',
    'JLT': '100', 'JNE': '101', 'JLE': '110', 'JMP': '111'
}

def load_asm() -> list:
```

```

"""
loads the assembly code and returns the code as filtered with no \n or
comments
"""
filtered_code = []
with open(asm_path, 'r') as asm:
    for line_no, line in enumerate(asm):
        # Removes \n and comments
        #filtered code is organized as a tuple where line[0] is line #
        # and line[1] is code
        line = line.split('//')[0].strip()
        if line:
            filtered_code.append([line_no, line])
    return filtered_code

def symbol_table(code: list) -> dict:
    """
    goes through filtered code and creating symbol table
    """
    symbol = {
        'R0': 0, 'R1': 1, 'R2': 2, 'R3': 3, 'R4': 4, 'R5': 5, 'R6': 6,
        'R7': 7, 'R8': 8, 'R9': 9, 'R10': 10, 'R11': 11, 'R12': 12,
        'R13': 13, 'R14': 14, 'R15': 15, 'SCREEN': 16384, 'KBD': 24576,
        'SP': 0, 'LCL': 1, 'ARG': 2, 'THIS': 3, 'THAT': 4
    }

    instruction_address = 0
    for line in code:
        #looking for loop names which start ( and end with )
        if line[1].startswith('(') and line[1].endswith(')'):
            label = line[1][1:-1] #removes ( )
            if label not in symbol:
                symbol[label] = instruction_address #adds symbol to symbol
            table
        else:
            instruction_address += 1 #goes to next memory spot
    #starts at 16 because 0-15 has predefined variables
    next_variable_address = 16
    for line in code:
        if line[1].startswith('@'):

```

```

        symbol_name = line[1][1:] #removing @
        #adds variable to symbol table by making sure its actually a
variable and not memory address
        if not symbol_name.isdigit() and symbol_name not in symbol: #if
not an integer or found in symbol table
            symbol[symbol_name] = next_variable_address
            next_variable_address += 1
    return symbol

def binary_code(code: list, symbol: dict):
    """
    generates binary code and writes it to .hack file
    """
    with open(bin_path, 'w') as bin_:
        #found this function online to generate twos-complement from base
10
        twos_complement = lambda num, bits: bin((num + (1 << bits)) % (1 <<
bits))[2:].zfill(bits)
        for line in code:
            #don't need to convert named loops again
            if not line[1] or line[1].startswith('('):
                continue
            elif line[1].startswith('@'):
                symbol_name = line[1][1:]
                #gets base 10 value that needs to be converted to binary
                #checks to see if first the symbol is a variable or
directly an integer already
                address = int(symbol[symbol_name]) if symbol_name in symbol
else int(symbol_name)
                #converts to 16 bit two's complement
                byt = twos_complement(address, 16)
                #writes to file
                bin_.write(f'{byt}\n')
            #labeling c-instruction
            else:
                if '=' in line[1]:
                    dest, remainder = line[1].split('=')
                else:
                    dest, remainder = 'null', line[1]
                if ';' in remainder:

```

```

        comp, jump = remainder.split(';')
    else:
        comp, jump = remainder, 'null'

    dest_1_code = dest_dict[dest]
    comp_1_code = comp_dict[comp]
    jump_1_code = jump_dict[jump]

    bin_code = '111' + comp_1_code + dest_1_code + jump_1_code
    bin_.write(f'{bin_code}\n')

if __name__ == '__main__':
    old_code = load_asm()
    code = copy.deepcopy(old_code)
    symbol = symbol_table(code)
    binary_code(old_code, symbol)








```

To test the assembler, each assembly code in Project 6 was converted into binary/hack code. To fully test the code, I ran the assembly and hack codes and saw if the two results agreed.

RESULTS OF Add.hack:

CPU Emulator (2.5) - C:\Users\kyleg\OneDrive\Desktop\CS271\nand2tetris_files\nand2tetris_files\02_software\nand2tetris\projects\06\add\Add.hack

File View Run Help

Animate: Program flow View: Screen Format: Decimal

Slow Fast

ROM	Bin
0	0000000000000010
1	1110110000010000
2	0000000000000011
3	1110000010010000
4	0000000000000000
5	1110001100001000
6	
7	
8	
9	
10	
11	
12	
13	
14	
15	
16	
17	
18	
19	
20	
21	
22	
23	
24	
25	
26	
27	
28	

RAM	
0	5
1	0
2	5
3	0
4	0
5	0
6	0
7	0
8	0
9	0
10	0
11	0
12	0
13	0
14	0
15	0
16	0
17	0
18	0
19	0
20	0
21	0
22	0
23	0
24	0
25	0
26	0
27	0
28	0

PC: 29 A: 0

D: 5

ALU

D Input: 5

M/A Input: 0

ALU output: 5

RESULTS OF Add.asm:

CPU Emulator (2.5) - C:\Users\kyleg\OneDrive\Desktop\CS271\nand2tetris_files\nand2tetris_files\02_software\nand2tetris\projects\06\add\Add.asm

File View Run Help

Slow Fast Animate: Program flow View: Screen Format: Decimal

ROM	Bin
0	0000000000000010
1	1110110000010000
2	0000000000000011
3	1110000010010000
4	0000000000000000
5	1110001100001000
6	
7	
8	
9	
10	
11	
12	
13	
14	
15	
16	
17	
18	
19	
20	
21	
22	
23	
24	
25	
26	
27	
28	

RAM	Bin
0	5
1	0
2	5
3	0
4	0
5	0
6	0
7	0
8	0
9	0
10	0
11	0
12	0
13	0
14	0
15	0
16	0
17	0
18	0
19	0
20	0
21	0
22	0
23	0
24	0
25	0
26	0
27	0
28	0

PC 43 A 0

D 5

ALU
D Input: 5
M/A Input: 0
ALU output: 5

RESULTS OF Max.hack:

CPU Emulator (2.5) - C:\Users\kyleg\OneDrive\Desktop\CS271\nand2tetris_files\nand2tetris_files\02_software\nand2tetris\projects\06\max\Max.hack

File View Run Help

Slow Fast Animate: Program flow View: Screen Format: Decimal

ROM	Bin
0	0000000000000000
1	1111110000010000
2	0000000000000001
3	1111010011010000
4	0000000000001010
5	1110001100000001
6	0000000000000001
7	1111110000010000
8	0000000000001100
9	1110101010000111
10	0000000000000000
11	1111110000010000
12	0000000000000010
13	1110001100001000
14	0000000000001110
15	1110101010000111
16	
17	
18	
19	
20	
21	
22	
23	
24	
25	
26	
27	
28	

RAM	Bin
0	5
1	0
2	5
3	0
4	0
5	0
6	0
7	0
8	0
9	0
10	0
11	0
12	0
13	0
14	0
15	0
16	0
17	0
18	0
19	0
20	0
21	0
22	0
23	0
24	0
25	0
26	0
27	0
28	0

PC 15 A 14

D 5

ALU
D Input: 5
M/A Input: 14
ALU output: 0

Results of Max.asm:

CPU Emulator (2.5) - C:\Users\kyleg\OneDrive\Desktop\CS271\nand2tetris_files\nand2tetris_files\02_software\nand2tetris\projects\06\max\Max.asm

File View Run Help

Slow Fast Animate: Program flow View: Screen Format: Decimal

ROM	Bin
0	0000000000000000
1	1111110000010000
2	0000000000000001
3	1111010011010000
4	0000000000001010
5	1110001100000001
6	0000000000000001
7	1111110000010000
8	0000000000001100
9	1110101010000111
10	0000000000000000
11	1111110000010000
12	0000000000000010
13	1110001100001000
14	0000000000001110
15	1110101010000111
16	
17	
18	
19	
20	
21	
22	
23	
24	
25	
26	
27	
28	

RAM	
0	5
1	0
2	5
3	0
4	0
5	0
6	0
7	0
8	0
9	0
10	0
11	0
12	0
13	0
14	0
15	0
16	0
17	0
18	0
19	0
20	0
21	0
22	0
23	0
24	0
25	0
26	0
27	0
28	0

PC: 15 A: 14

D: 5

ALU

D Input: 5

M/A Input: 14

ALU output: 0

RESULTS OF MaxL.hack:

CPU Emulator (2.5) - C:\Users\kyleg\OneDrive\Desktop\CS271\nand2tetris_files\nand2tetris_files\02_software\nand2tetris\projects\06\max\MaxL.hack

File View Run Help

Slow Fast Animate: Program flow View: Screen Format: Decimal

ROM	Bin
0	0000000000000000
1	1111110000010000
2	0000000000000001
3	1111010011010000
4	0000000000001010
5	1110001100000001
6	0000000000000001
7	1111110000010000
8	0000000000001100
9	1110101010000111
10	0000000000000000
11	1111110000010000
12	0000000000000010
13	1110001100001000
14	0000000000001110
15	1110101010000111
16	
17	
18	
19	
20	
21	
22	
23	
24	
25	
26	
27	
28	

RAM	
0	5
1	0
2	5
3	0
4	0
5	0
6	0
7	0
8	0
9	0
10	0
11	0
12	0
13	0
14	0
15	0
16	0
17	0
18	0
19	0
20	0
21	0
22	0
23	0
24	0
25	0
26	0
27	0
28	0

PC: 14 A: 14

D: 5

ALU: D Input: 5, M/A Input: 14, ALU output: 0

RESULTS OF MaxL.asm:

CPU Emulator (2.5) - C:\Users\kyleg\OneDrive\Desktop\CS271\nand2tetris_files\nand2tetris_files\02_software\nand2tetris\projects\06\max\MaxL.asm

File View Run Help

Slow Fast Animate: Program flow View: Screen Format: Decimal

ROM	Bin
0	0000000000000000
1	1111110000010000
2	0000000000000001
3	1111010011010000
4	0000000000001010
5	1110001100000001
6	0000000000000001
7	1111110000010000
8	0000000000001100
9	1110101010000111
10	0000000000000000
11	1111110000010000
12	0000000000000010
13	1110001100001000
14	0000000000001110
15	1110101010000111
16	
17	
18	
19	
20	
21	
22	
23	
24	
25	
26	
27	
28	

RAM	
0	5
1	0
2	5
3	0
4	0
5	0
6	0
7	0
8	0
9	0
10	0
11	0
12	0
13	0
14	0
15	0
16	0
17	0
18	0
19	0
20	0
21	0
22	0
23	0
24	0
25	0
26	0
27	0
28	0

PC: 15 A: 14

D: 5

ALU: D Input: 5, M/A Input: 14, ALU output: 0

RESULTS OF Pong.hack:

CPU Emulator (2.5) - C:\Users\kyleg\OneDrive\Desktop\CS271\nand2tetris_files\nand2tetris_files\02_software\nand2tetris\projects\06\pong\Pong.hack

File View Run Help

Slow Fast Animate: No animation View: Screen Format: Decimal

ROM	Bin
0	0000000100000000
1	1110110000010000
2	0000000000000000
3	1110001100001000
4	0000000010000101
5	1110101010000111
6	0000000000001111
7	1110001100001000
8	0000000000000000
9	1111110010101000
10	1111110000010000
11	1110110010100000
12	1111000111010000
13	1110101010001000
14	0000000000010011
15	1110001100000101
16	0000000000000000
17	1111110010100000
18	1110111010001000
19	0000000000001111
20	1111110000100000
21	1110101010000111
22	0000000000001111
23	1110001100001000
24	0000000000000000
25	1111110010101000
26	1111110000010000
27	1110110010100000
28	1111000111010000

RAM	
0	5
1	0
2	5
3	0
4	0
5	0
6	0
7	0
8	0
9	0
10	0
11	0
12	0
13	0
14	0
15	0
16	0
17	0
18	0
19	0
20	0
21	0
22	0
23	0
24	0
25	0
26	0
27	0
28	0

PC: 0 A: 0

Game Over

Score: 0

D: 0

ALU

D Input: 0 M/A Input: 0 ALU output: 0

Running...

RESULTS OF Pong.asm:

CPU Emulator (2.5) - C:\Users\kyleg\OneDrive\Desktop\CS271\nand2tetris_files\nand2tetris_files\02_software\nand2tetris\projects\06\pong\Pong.asm

File View Run Help

Slow Fast Animate: No animation View: Screen Format: Decimal

ROM	Bin
0	0000000100000000
1	1110110000010000
2	0000000000000000
3	1110001100001000
4	0000000010000101
5	1110101010000111
6	0000000000001111
7	1110001100001000
8	0000000000000000
9	1111110010101000
10	1111110000010000
11	1110110010100000
12	1111000111010000
13	1110101010001000
14	0000000000010011
15	1110001100000101
16	0000000000000000
17	1111110010100000
18	1110111010001000
19	0000000000001111
20	1111110000100000
21	1110101010000111
22	0000000000001111
23	1110001100001000
24	0000000000000000
25	1111110010101000
26	1111110000010000
27	1110110010100000
28	1111000111010000

RAM	
0	267
1	266
2	261
3	0
4	0
5	0
6	0
7	0
8	0
9	0
10	0
11	0
12	0
13	0
14	27177
15	9717
16	4872
17	2050
18	2068
19	0
20	16384
21	-1
22	3570
23	18
24	2105
25	2118
26	3495
27	16384
28	-1

PC: 0 A: 0

Score: 1

D: 0

ALU

D Input: 0 M/A Input: 0 ALU output: 0

Running...

RESULTS OF PongL.hack:

CPU Emulator (2.5) - C:\Users\kyleg\OneDrive\Desktop\CS271\nand2tetris_files\nand2tetris_files\02_software\nand2tetris\projects\06\pong\PongL.hack

File View Run Help

Slow Fast Animate: No animation View: Screen Format: Decimal

ROM	Bin
0	0000000100000000
1	1110110000010000
2	0000000000000000
3	1110001100001000
4	0000000010000101
5	1110101010000111
6	0000000000001111
7	1110001100001000
8	0000000000000000
9	1111110010101000
10	1111110000010000
11	1110110010100000
12	1111000111010000
13	1110101010001000
14	0000000000010011
15	1110001100000101
16	0000000000000000
17	1111110010100000
18	1110111010001000
19	0000000000001111
20	1111110000100000
21	1110101010000111
22	0000000000001111
23	1110001100001000
24	0000000000000000
25	1111110010101000
26	1111110000010000
27	1110110010100000
28	1111000111010000

RAM	
0	267
1	266
2	261
3	0
4	0
5	0
6	0
7	0
8	0
9	0
10	0
11	0
12	0
13	0
14	27177
15	9717
16	4872
17	2050
18	2068
19	0
20	16384
21	-1
22	3570
23	18
24	2105
25	2118
26	3495
27	16384
28	-1

PC: 0 A: 0

Score: 0

D: 0

ALU

D Input: 0

M/A Input: 0

ALU output: 0

Running...

RESULTS OF PongL.asm:

CPU Emulator (2.5) - C:\Users\kyleg\OneDrive\Desktop\CS271\nand2tetris_files\nand2tetris_files\02_software\nand2tetris\projects\06\pong\PongL.asm

File View Run Help

Slow Fast Animate: No animation View: Screen Format: Decimal

ROM	Bin
0	0000000100000000
1	1110110000010000
2	0000000000000000
3	1110001100001000
4	0000000010000101
5	1110101010000111
6	0000000000001111
7	1110001100001000
8	0000000000000000
9	1111110010101000
10	1111110000010000
11	1110110010100000
12	1111000111010000
13	1110101010001000
14	0000000000010011
15	1110001100000101
16	0000000000000000
17	1111110010100000
18	1110111010001000
19	0000000000001111
20	1111110000100000
21	1110101010000111
22	0000000000001111
23	1110001100001000
24	0000000000000000
25	1111110010101000
26	1111110000010000
27	1110110010100000
28	1111000111010000

RAM	
0	266
1	266
2	261
3	0
4	0
5	0
6	0
7	0
8	0
9	0
10	0
11	0
12	0
13	0
14	27177
15	9717
16	4872
17	2050
18	2068
19	0
20	16384
21	-1
22	3570
23	18
24	2105
25	2118
26	3495
27	16384
28	-1

PC: 0 A: 0

Score: 0

D: 0

ALU

D Input: 0

M/A Input: 0

ALU output: 0

RESULTS of Rect.hack:

CPU Emulator (2.5) - C:\Users\kyleg\OneDrive\Desktop\CS271\nand2tetris_files\nand2tetris_files\02_software\nand2tetris\projects\06\rect\Rect.hack

File View Run Help

Slow Fast Animate: Program flow View: Screen Format: Decimal

ROM	Bin
0	0000000000000000
1	1111110000010000
2	0000000000010111
3	1110001100000110
4	0000000000010000
5	1110001100001000
6	0100000000000000
7	1110110000010000
8	0000000000010001
9	1110001100001000
10	0000000000010001
11	1111110000010000
12	1110110100010000
13	0000000000010001
14	1111110000010000
15	0000000000010000
16	1110000010010000
17	0000000000010001
18	1110001100001000
19	0000000000010000
20	1111110010011000
21	000000000001010
22	1110001100000001
23	0000000000010111
24	1110101010000111
25	
26	
27	
28	

RAM	
2	0
3	0
4	0
5	0
6	0
7	0
8	0
9	0
10	0
11	0
12	0
13	0
14	0
15	0
16	0
17	16544
18	0
19	0
20	0
21	0
22	0
23	0
24	0
25	0
26	0
27	0
28	0
29	0
30	0

PC 24 A 23

D 0

ALU

D Input: 0 M/A Input: 23 ALU output: 0

RESULTS of rect.asm:

CPU Emulator (2.5) - C:\Users\kyleg\OneDrive\Desktop\CS271\nand2tetris_files\nand2tetris_files\02_software\nand2tetris\projects\06\rect\Rect.asm

File View Run Help

Slow Fast Animate: Program flow View: Screen Format: Decimal

ROM	Bin
0	0000000000000000
1	1111110000010000
2	0000000000010111
3	1110001100000110
4	0000000000010000
5	1110001100001000
6	0100000000000000
7	1110110000010000
8	0000000000010001
9	1110001100001000
10	0000000000010001
11	1111110000010000
12	1110110100010000
13	0000000000010001
14	1111110000010000
15	0000000000010000
16	1110000010010000
17	0000000000010001
18	1110001100001000
19	0000000000010000
20	1111110010011000
21	000000000001010
22	1110001100000001
23	0000000000010111
24	1110101010000111
25	
26	
27	
28	

RAM	
2	0
3	0
4	0
5	0
6	0
7	0
8	0
9	0
10	0
11	0
12	0
13	0
14	0
15	0
16	0
17	16544
18	0
19	0
20	0
21	0
22	0
23	0
24	0
25	0
26	0
27	0
28	0
29	0
30	0

PC 24 A 23

D 0

Screen

ALU

D Input: 0 M/A Input: 23 ALU output: 0

RESULTS of RectL.hack:

CPU Emulator (2.5) - C:\Users\kyleg\OneDrive\Desktop\CS271\nand2tetris_files\nand2tetris_files\02_software\nand2tetris\projects\06\rect\RectL.hack

File View Run Help

Slow Fast Animate: Program flow View: Screen Format: Decimal

ROM	Bin
0	0000000000000000
1	1111110000010000
2	0000000000010111
3	1110001100000110
4	0000000000010000
5	1110001100001000
6	0100000000000000
7	1110110000010000
8	0000000000010001
9	1110001100001000
10	0000000000010001
11	1111110000100000
12	1110110100001000
13	0000000000010001
14	1111110000010000
15	0000000001000000
16	1110000010010000
17	0000000000010001
18	1110001100001000
19	0000000000010000
20	1111110010011000
21	0000000000001010
22	1110001100000001
23	0000000000010111
24	1110101010000111
25	
26	
27	
28	

RAM	
2	0
3	0
4	0
5	0
6	0
7	0
8	0
9	0
10	0
11	0
12	0
13	0
14	0
15	0
16	0
17	16544
18	0
19	0
20	0
21	0
22	0
23	0
24	0
25	0
26	0
27	0
28	0
29	0
30	0

PC: 24 A: 23

D: 0

ALU: D Input: 0 M/A Input: 23 ALU output: 0

RESULTS of RectL.asm:

CPU Emulator (2.5) - C:\Users\kyleg\OneDrive\Desktop\CS271\nand2tetris_files\nand2tetris_files\02_software\nand2tetris\projects\06\rect\RectL.asm

File View Run Help

Slow Fast Animate: Program flow View: Screen Format: Decimal

ROM	Bin
0	0000000000000000
1	1111110000010000
2	0000000000010111
3	1110001100000110
4	0000000000010000
5	1110001100001000
6	0100000000000000
7	1110110000010000
8	0000000000010001
9	1110001100001000
10	0000000000010001
11	1111110000100000
12	1110110100001000
13	0000000000010001
14	1111110000010000
15	0000000001000000
16	1110000010010000
17	0000000000010001
18	1110001100001000
19	0000000000010000
20	1111110010011000
21	0000000000001010
22	1110001100000001
23	0000000000010111
24	1110101010000111
25	
26	
27	
28	

RAM	
2	0
3	0
4	0
5	0
6	0
7	0
8	0
9	0
10	0
11	0
12	0
13	0
14	0
15	0
16	0
17	16544
18	0
19	0
20	0
21	0
22	0
23	0
24	0
25	0
26	0
27	0
28	0
29	0
30	0

PC: 23 A: 23

D: 0

ALU: D Input: 0 M/A Input: 23 ALU output: 0