CS 220 Lab #08 – Virtual Machine Part 1

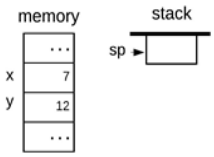
Name: Alexandra Burke

Date: _____

1. Stack Arithmetic Commands: What the state of the stack and the memory after the following VM code is executed. Where will the stack pointer (sp) end up, if it originally begins at address 256? Please illustrate the stack after every VM command has been executed.

VM code

```
push 2
push x
sub
push y
push 9
add
add
```

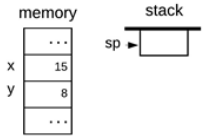


	memory:	stack:
push 2	|__ | 7 | 12 | __ | x y	| 2 | sp=2
push x	|__ | 7 | 12 | __ | x y	| 2 | 7 | sp=7
sub	|__ | 5 | 12 | __ | x y	| 5 | sp=5
push y	|__ | 5 | 12 | __ | x y	| 5 | 12 | sp=12
push 9	|__ | 5 | 12 | __ | x y	| 5 | 12 | 9 | sp=9
add	|__ | 5 | 21 | __ | x y	| 5 | 21 | sp=21
add	|__ | 5 | 26 | __ | x y	| 26 | sp=26

2. Stack Logical Commands: Again, what the state of the stack and memory after the following VM code is executed. Where will the stack pointer (sp) end up, if it originally begins at address 256? Please illustrate the stack after every VM command has been executed.

VM code

```
push x
push 7
lt
push y
push 8
eq
or
```



	memory:	stack:
push x	|__ | 256 | 257 | __ | x y	| x | sp=256
push 7	| 255 | 256 | 257 |	| 7 | x | sp=7
lt	| 254 | 255 | 256 | 257 |	| 1 | x | 7 | sp=1
push y	| 253 | 254 | 255 | 256 | 257 |	| y | 1 | x | 7 | sp= y
push 8	| 252| 253 | 254 | 255 | 256 | 257 |	| 8 | y | 1 | x | 7 | sp=8
eq	| 251 | 252| 253 | 254 | 255 | 256 | 257 |	| 0 | y | 8 | 1| x | 7| sp=0
or	|250|251|252|253|254|255|256|257|	| 1 | y | 8 | 1 | x | 7 | sp=1

3. Suppose the state of the argument and local memory segments are as follows:

	argument
0	9
1	14

	local
0	
1	?
2	

Now consider the following VM code:

```
1   push constant 0
2   pop local 0
3   push constant 15
4   pop local 1
5   push local 1
6   push argument 1
7   gt
8   pop local 2
9   push local 0
10  push argument 0
11  add
12  pop local 0
13  push local 1
14  push local 1
15  push constant 1
16  sub
17  add
18  pop local 1
```

What will be the value of local 1 after the VM code has executed?_____29_____

4. Suppose the state of the RAM is as follows and the adjacent assembly code will execute:

	RAM
0	3
1	2
2	0
3	6
4	5
5	1
6	4

```
1   @1
2   A = M
3   A = M
4   A = M
5   D = M
6   @4
7   M = D
```

What will be the value of the RAM[4] following the assembly code execution?____RAM[4] = 6_____

5. Suppose the state of the RAM is as follows and the adjacent assembly code will execute:

	RAM
0	3
1	2
2	0
3	6
4	5
5	1
6	4

```
1  @5
2  A=M
3  A=A+1
4  A=A+1
5  D=M
6  A=A+1
7  M=D
```

What will be the value of the RAM[4] following the assembly code execution?_____0_____

6. Suppose the state of the RAM is as follows and the adjacent pseudocode (like C++) will execute:

	RAM
256	22
257	31
258	200
259	28

```
1  p1 = 256
2  p1 = p1 + 3
3  *p1 = *p1 + 3
4  p2 = p1 - 2
5  p1--
6  *(p2 + 1) = *p1 + *p2
```

What will be the value of the RAM[258] following the assembly code execution?_____231_____

Translate the following VM commands to <u>Assembly</u> instructions:

- ❑ push constant 1

```
@1
D=A
@SP
A=M
M=D
@SP
M=M+1
```

- ❑ push constant 5

```
@5
D=A
@SP
A=M
M=D
@SP
M=M+1
```

- ❑ add

```
@SP
M=M-1
A=M
D=M
@SP
M=M-1
A=M
M=M+D
@SP
M=M+1
```

- ❑ pop static 7 //suppose inside of a file named **Add**

```
@SP
M=M-1

A=M
D=M

@Add.7
M=D
```

- ❑ pop local 2

```
// pop local 2

@SP
M=M-1
A=M
D=M
@LCL   // Load the base address of the local segment (LCL) into the A-register
D=M+2  // Add 2 to the base address to get the target memory address
M=D
```

- ❑ eq

```
// SP points to the next available location on the stack
@SP
M=M-1
A=M
D=M
@SP
M=M-1
A=M
D=M-D
@EQUAL_LABEL
D;JEQ // Jump to the EQUAL_LABEL if D is equal to 0 (x == y)
D=0
@END_LABEL
0;JMP // Unconditional jump to the END_LABEL
(EQUAL_LABEL)
D=-1
// Push the result (D) onto the stack
@SP
A=M
M=D
@SP
M=M+1
(END_LABEL)
```