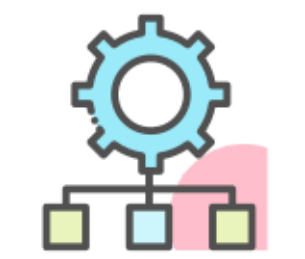


```
push constant 7
push constant 8
add
```



VM Translator

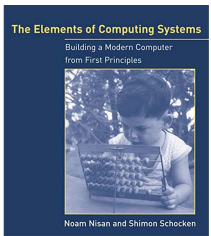
```
@7
D=A
@SP
AM=M+1
A=A-1
M=D

@8
D=A
@SP
AM=M+1
A=A-1
M=D

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

(END)
@END
0; JMP
```

Assembly instructions



Chap 7

VM to Hack Translator

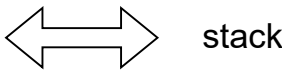
9 arithmetic / logical commands

```
1. add
2. sub
3. neg
4. eq
5. gt
6. lt
7. and
8. or
9. not
```

2 memory access commands, 8 memory segments

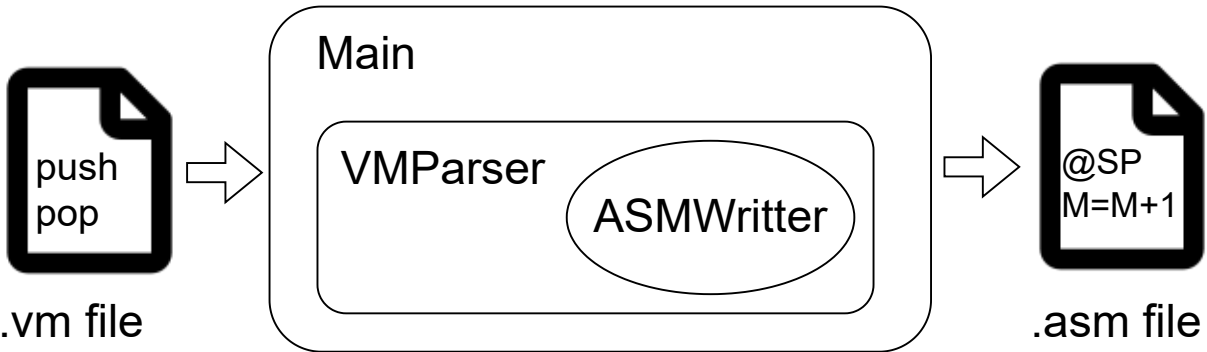
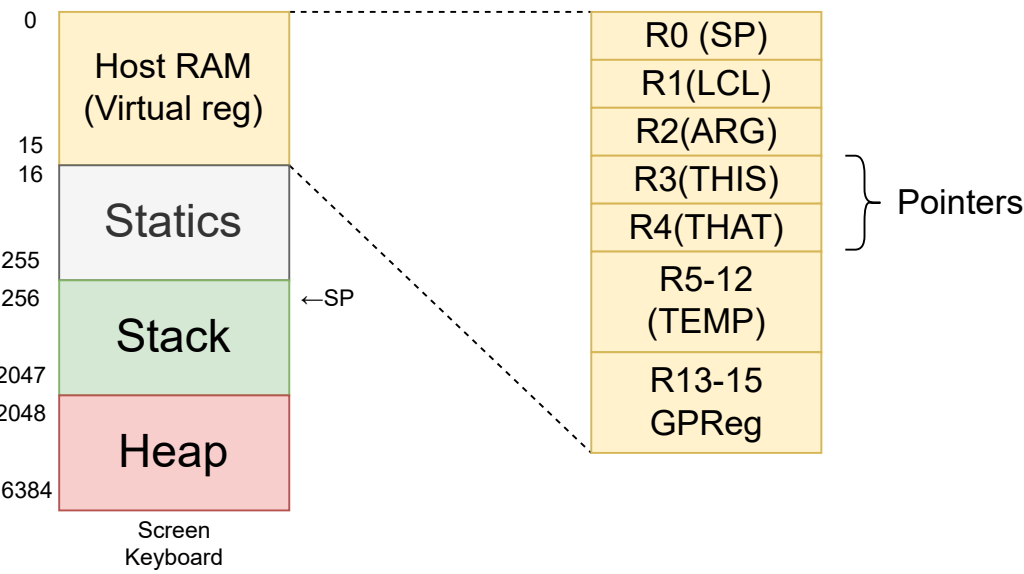
```
1. push
2. pop
```

```
1. constant
2. static
3. local
4. argument
5. pointer
6. temp
7. this
8. that
```



stack

Hack virtual machine segments

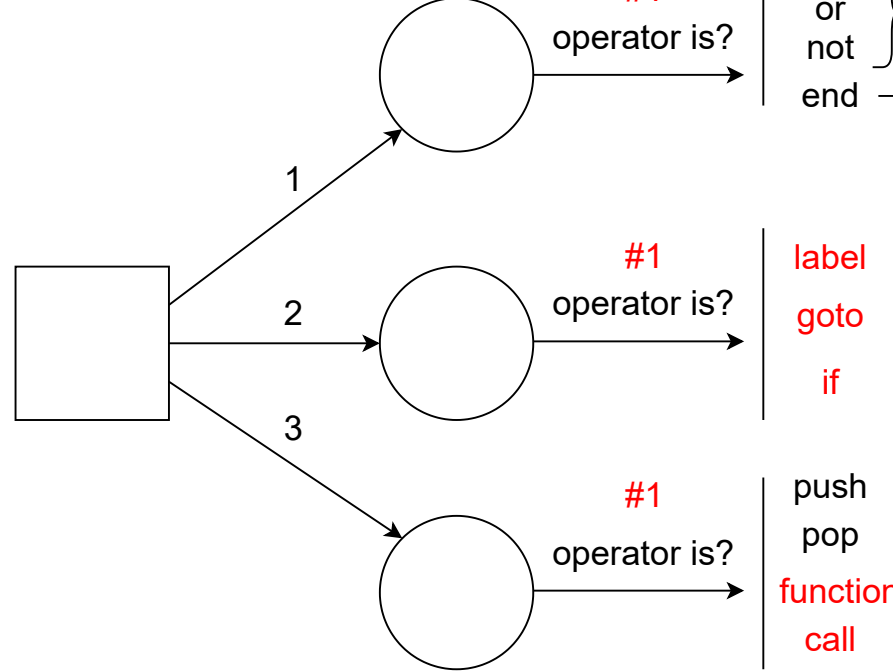
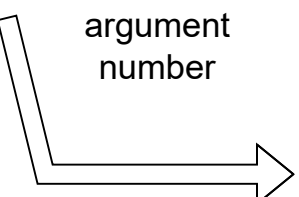
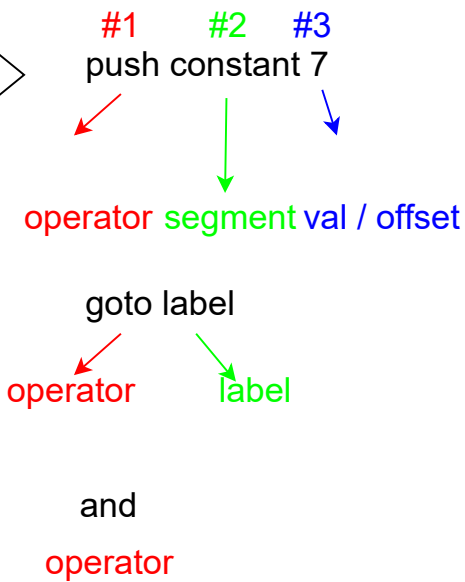


Main: read .vm file line by line, write translated assembly code into .asm file.

VMPParser: Break the command in each line into operation, memory segment and offsets (or constant value), then call the asm code writer functions in ASMWritter.

ASMWritter: generates the assembly code based on the called function by the VMPParser and the parameters passed in.

```
push constant 7
push constant 8
add
```



Red operations are covered in project 08

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

sub    @SP
      AM=M-1
      D=M
      A=A-1
      M=M-D

neg    @SP
      A=M
      A=A-1
      M=-M
```

```
eq    @SP
      AM=M-1
      D=M
      @SP
      AM=M-1
      D=M-D
      @labelTrue
      D;JEQ
      D=0
      @labelFalse
      0;JMP
      (labelTrue)
      D=-1
      (labelFalse)

gt    @SP
      AM=M-1
      D=M
      @SP
      AM=M-1
      D=M-D
      @labelTrue
      D;JGT
      D=0
      @labelFalse
      0;JMP
      (labelTrue)
      D=-1
      (labelFalse)

lt    @SP
      AM=M-1
      D=M
      @SP
      AM=M-1
      D=M-D
      @labelTrue
      D;JLT
      D=0
      @labelFalse
      0;JMP
      (labelTrue)
      D=-1
      (labelFalse)
```

```
and    @SP
      AM=M-1
      D=M
      A=A-1
      M=D&M

or     @SP
      AM=M-1
      D=M
      A=A-1
      M=D|M

not    @SP
      A=M
      A=A-1
      M=!M
```

```
end
(END)
0; JMP
```

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

Error, cannot
pop to constant!
```

```
@segment
D=M
@offset
A=D+A
D=M
@SP
A=M
M=D
@SP
AM=M-1
D=M
@R13
A=M
M=D
```

```
@segment
D=M
@offset
D=D+A
@R13
M=D
@SP
AM=M-1
D=M
@R13
A=M
M=D
```



Github



Youtube