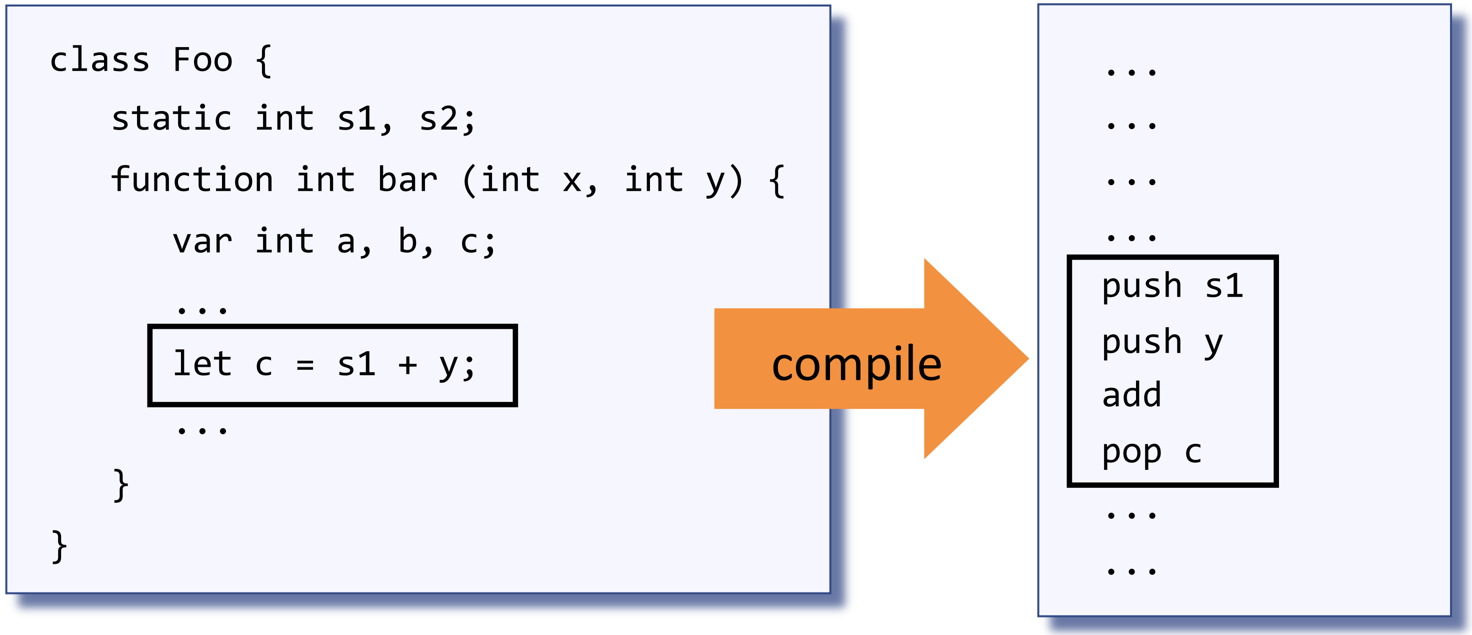
The big picture



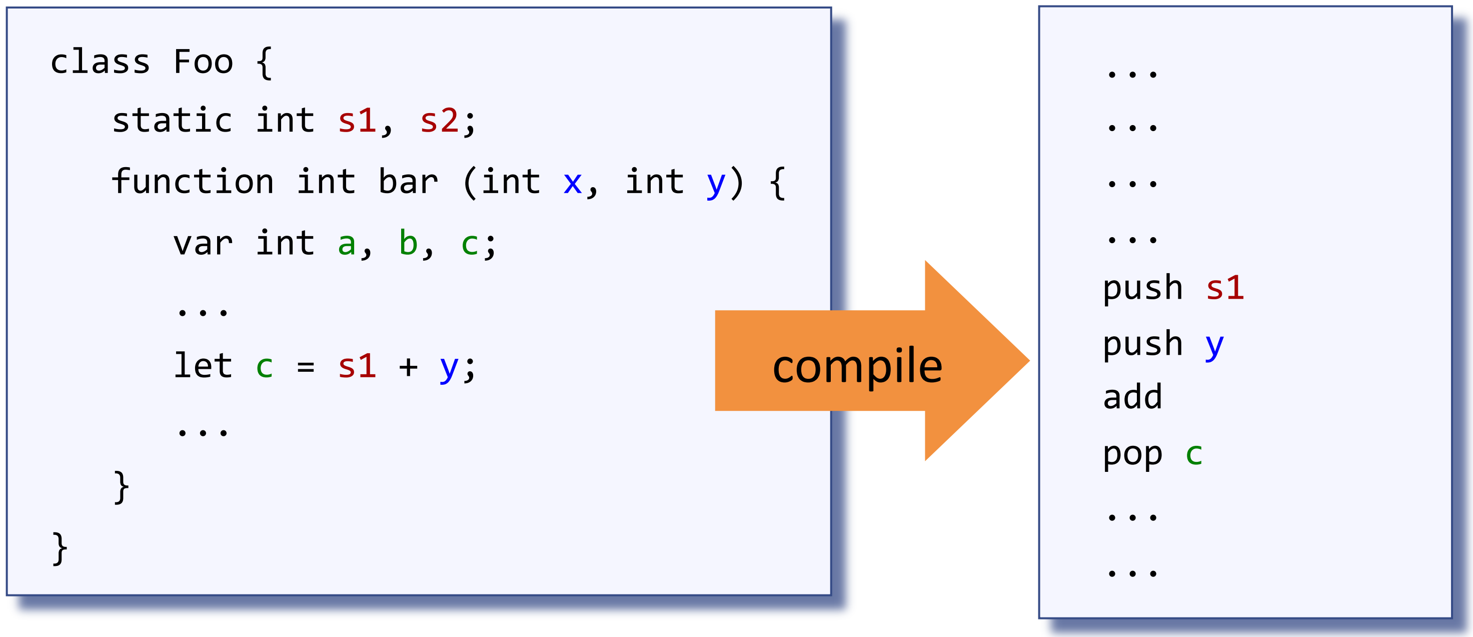
Source code (Jack)

```
class Foo {
   static int s1, s2;
   function int bar (int x, int y) {
      var int a, b, c;
      ...
      let c = s1 + y;
      ...
   }
}
```

Compiled VM code

```
...
...
...
...
push s1
push y
add
pop c
...
...
```

Variable kinds



Source code (Jack)

```
class Foo {
   static int s1, s2;
   function int bar (int x, int y) {
      var int a, b, c;
      ...
      let c = s1 + y;
      ...
   }
}
```

Compiled VM code

```
...
...
...
...
push s1
push y
add
pop c
...
...
```

Variable kinds

- Argument variables
- Local variables
- Static variables

(More kinds later)

Variable kinds and memory segments

Source code (Jack)

```
class Foo {
   static int s1, s2;
   function int bar (int x, int y) {
      var int a, b, c;
      ...
      let c = s1 + y;
      ...
   }
}
```

compile

Compiled VM code

```
...
...
...
...
push s1
push y
add
pop c
...
...
```

Virtual memory segments:

	argument	
0		x
1		y
2		
3		
...	...	

	local	
0		a
1		b
2		c
3		
...	...	

	static	
0		s1
1		s2
2		
3		
...	...	

Variable kinds and memory segments

Source code (Jack)

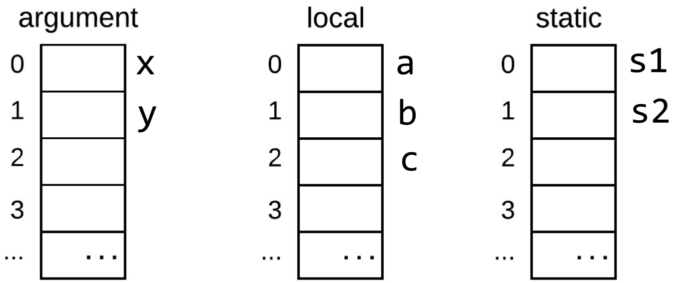
```
class Foo {
   static int s1, s2;
   function int bar (int x, int y) {
      var int a, b, c;
      ...
      let c = s1 + y;
      ...
   }
}
```

compile

Compiled VM code

```
...
...
...
...
push static 0
push y
add
pop c
...
...
```

Virtual memory segments:



argument			local			static		
0		x	0		a	0		s1
1		y	1		b	1		s2
2			2		c	2		
3			3			3		
...	...		...	...		...	...	

Variable kinds and memory segments

Source code (Jack)

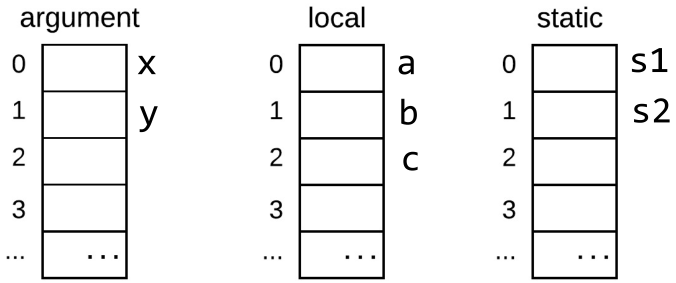
```
class Foo {
   static int s1, s2;
   function int bar (int x, int y) {
      var int a, b, c;
      ...
      let c = s1 + y;
      ...
   }
}
```

compile

Compiled VM code

```
...
...
...
...
push static 0
push argument 1
add
pop c
...
...
```

Virtual memory segments:



	argument			local			static	
0		x	0		a	0		s1
1		y	1		b	1		s2
2			2		c	2		
3			3			3		
...	...		...	...		...	...	

Variable kinds and memory segments

Source code (Jack)

```
class Foo {
   static int s1, s2;
   function int bar (int x, int y) {
      var int a, b, c;
      ...
      let c = s1 + y;
      ...
   }
}
```

compile

Compiled VM code

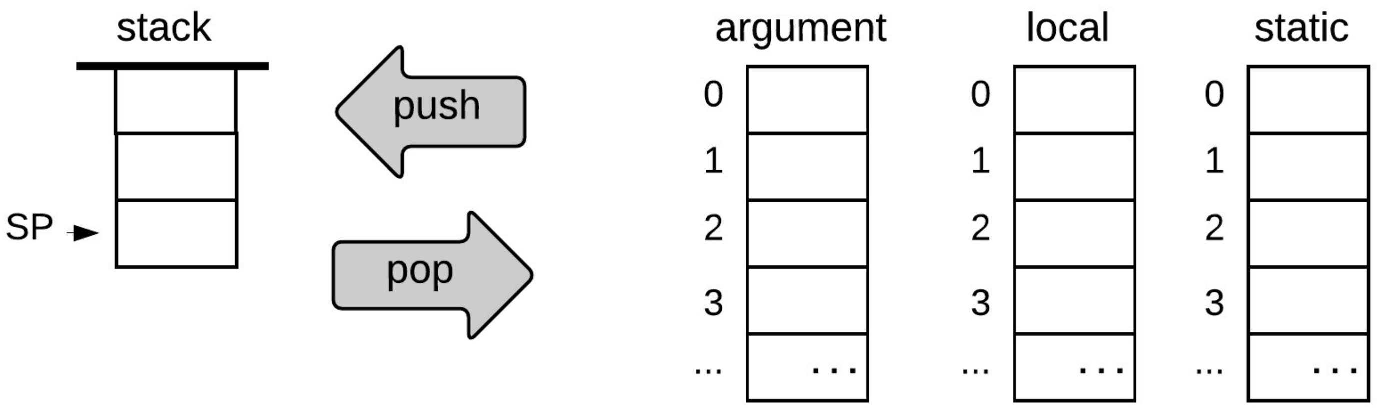
```
...
...
...
...
push static 0
push argument 1
add
pop local 2
...
...
```

Following compilation, all the symbolic references are replaced with references to virtual memory segments

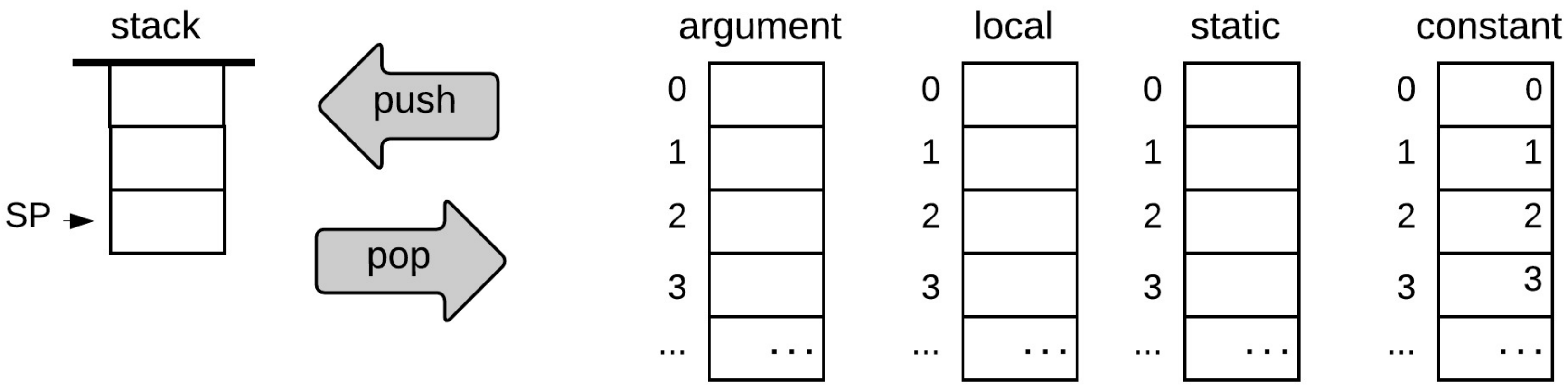
Virtual memory segments:

	argument		local		static
0	x	0	a	0	s1
1	y	1	b	1	s2
2		2	c	2	
3		3		3	
...	...	...	...	...	...

Memory segments



Memory segments

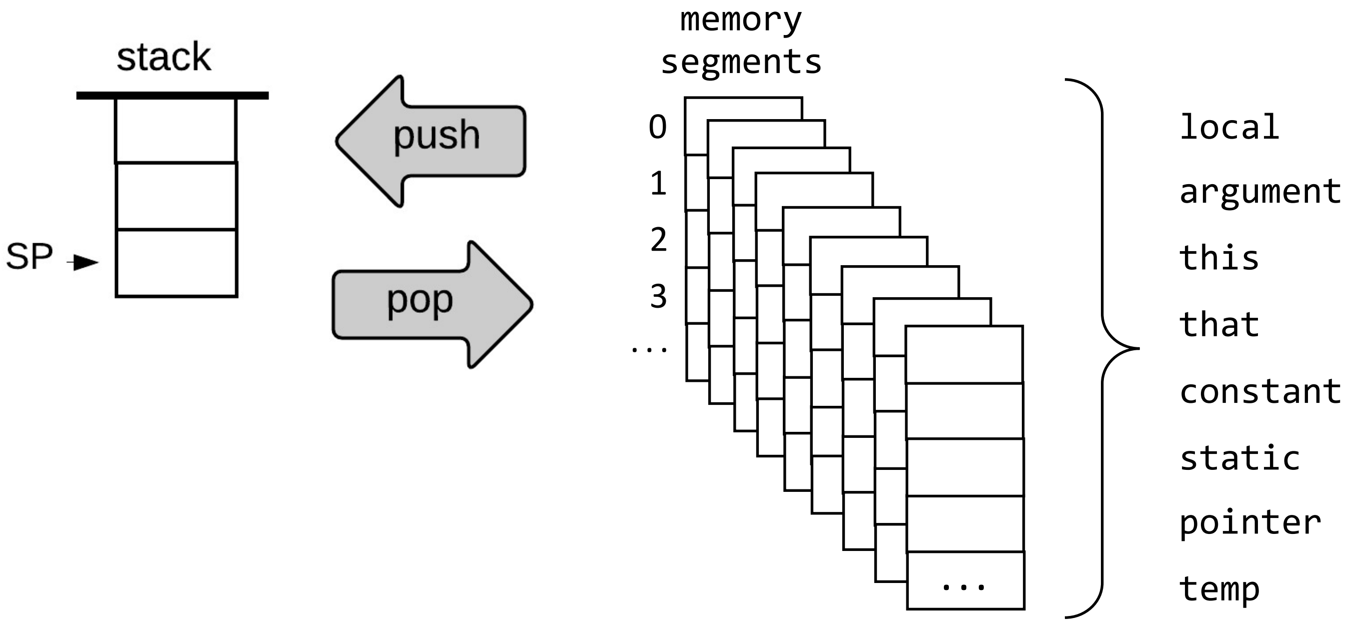


Syntax: `push` / `pop` *segment i*

Examples:

- `push constant 17`
- `pop local 2`
- `pop static 5`
- `push argument 3`
- `...`

Memory segments



<u>Syntax</u>: `push` *segment i*

Where *segment* is: `argument`, `local`, `static`, `constant`,
`this`, `that`, `pointer`, or `temp`

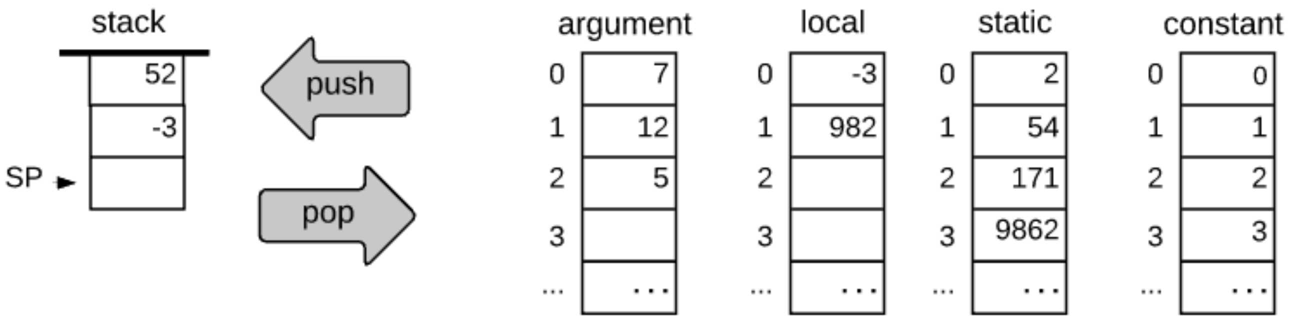
and *i* is a non-negative integer.

<u>Syntax</u>: `pop` *segment i*

Where *segment* is: `argument`, `local`, `static`,
`this`, `that`, `pointer`, or `temp`

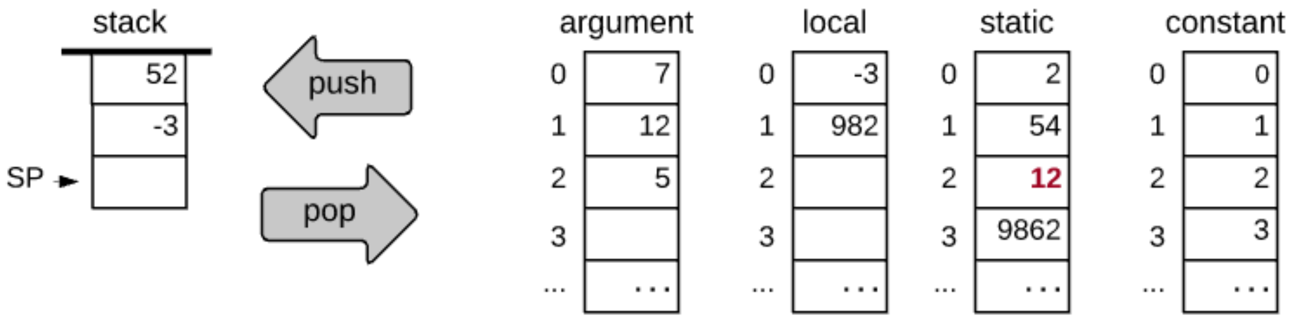
and *i* is a non-negative integer.

Quiz: write the missing code

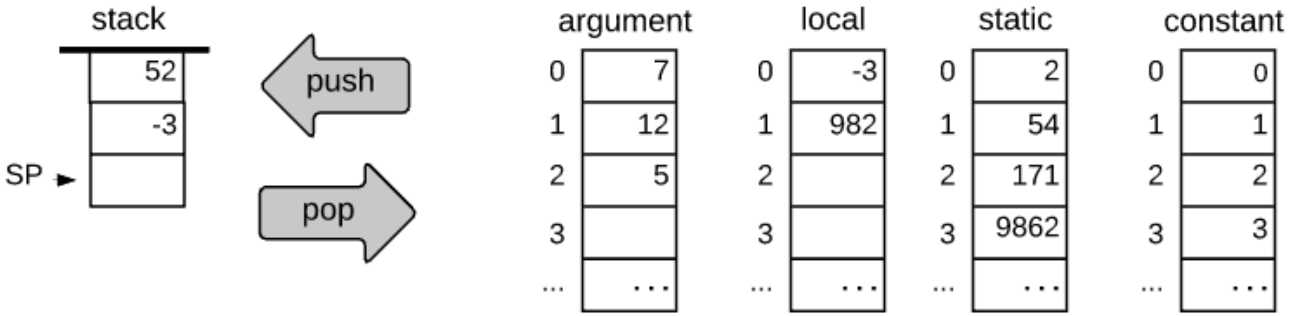


let `static 2 = argument 1`

?



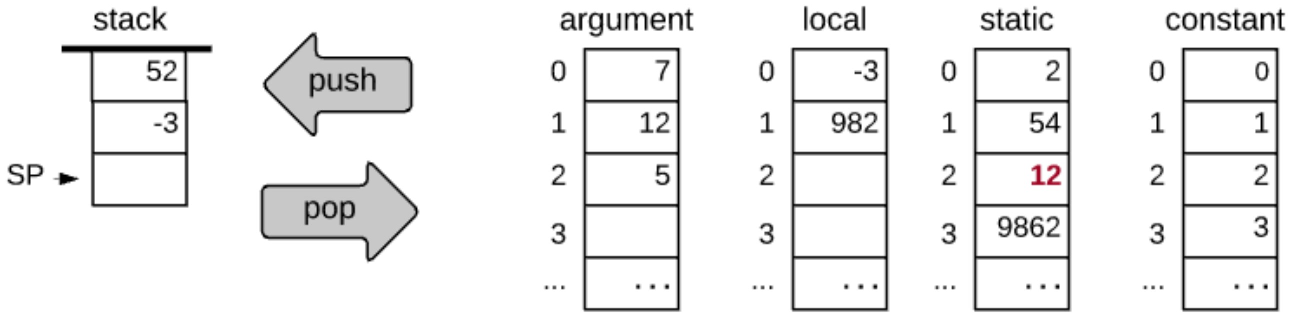
Quiz: write the missing code



	argument	local	static	constant
0	7	-3	2	0
1	12	982	54	1
2	5		171	2
3			9862	3
...	...	...	...	...

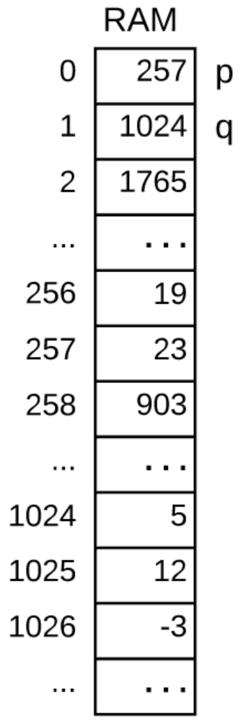
let `static 2 = argument 1`

```
push argument 1
pop static 2
```



	argument	local	static	constant
0	7	-3	2	0
1	12	982	54	1
2	5		**12**	2
3			9862	3
...	...	...	...	...

Pointer manipulation



RAM

0	257	p
1	1024	q
2	1765	
...	...	
256	19	
257	23	
258	903	
...	...	
1024	5	
1025	12	
1026	-3	
...	...	