# peek and isEmpty

- Property we want: If there has been a push and no pop, then isEmpty returns false
- With **peek** as written, property can be violated how?

```
Thread 1 (peek)

E ans = pop();

push(ans);

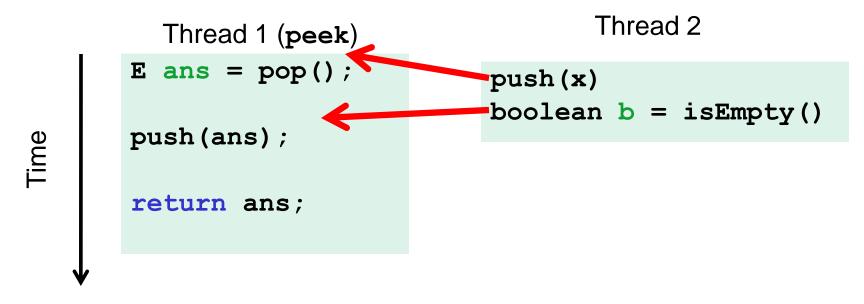
return ans;
```

Thread 2

push(x)
boolean b = isEmpty()

# peek and isEmpty

- Property we want: If there has been a push and no pop, then isEmpty returns false
- With **peek** as written, property can be violated how?



### peek and push

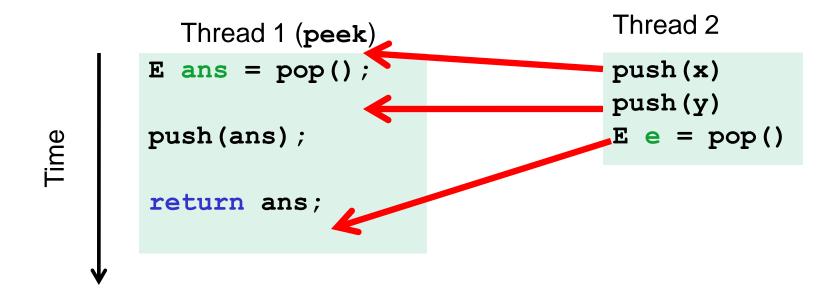
- Property we want: Values are returned from **pop** in LIFO order
- With **peek** as written, property can be violated how?

```
Thread 1 (peek)
E ans = pop();
push(ans);
return ans;
```

```
Thread 2
push(x)
push(y)
E e = pop()
```

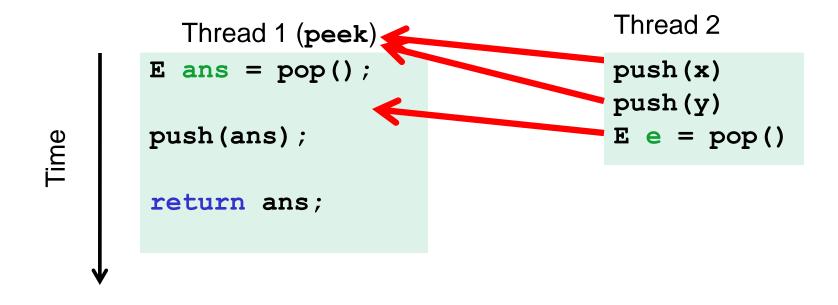
#### peek and push

- Property we want: Values are returned from **pop** in LIFO order
- With **peek** as written, property can be violated how?



## peek and pop

- Property we want: Values are returned from **pop** in LIFO order
- With **peek** as written, property can be violated how?



### peek and peek

- Property we want: peek does not throw an exception if number of pushes exceeds number of pops
- With **peek** as written, property can be violated how?

	Thread 1 (peek)	Thread 2
ime	E ans = pop();	E ans = pop();
	<pre>push(ans);</pre>	<pre>push(ans);</pre>
	<pre>return ans;</pre>	<pre>return ans;</pre>
N		

### peek and peek

- Property we want: peek doesn't throw an exception if number of pushes exceeds number of pops
- With **peek** as written, property can be violated how?

