```
void flipStack2 (StackOfT& s)
//! updates s
//! requires |s| = 2
//! ensures s = #s[1,2) * #s[0,1)
```

Name: _____

Name: _____

One CM: _____

Reasoning Table for *flipStack2*

S	Code	Assume	Confirm
0		|s0| = 2	true
	T y; StackOfT t;	/ / / / / / / / / / /	/ / / / / /
1		T.Init(y1) ^ t1 = <> ^ s1 = s0	s1 /= <>
	s.pop(y)	/ / / / / / / / / / /	/ / / / / /
2		<y2> is prefix of s1 ^ s2 = s1[1, |s1|) ^ t2 = t1	true
	t.push(y);	/ / / / / / / / / / /	/ / / / / /
3		t3 = <y2> * t2 ^ s3 = s2 ^ T.Init(y3)	s3 /= <>
	s.pop(y)	/ / / / / / / / / / /	/ / / / / /
4		<y4> is prefix of s3 ^ s4 = s3[1, |s3|) ^ t4 = t3	true
	t.push(y)	/ / / / / / / / / / /	/ / / / / /
5		t5 = <y4> * t4 ^ s5 = s4 ^ T.Init(y5)	true
	s.transferFrom(t);	/ / / / / / / / / / /	/ / / / / /
6		s6 = t5 ^ t6 = <> ^ y6 = y5	s6 = s0[1,2) * s0[0,1)

VC1: |s0| = 2 ^ T.Init(y1) ^ t1 = <> ^ s1 = s0 → s1 /= <>

VC3: |s0| = 2 ^ T.Init(y1) ^ t1 = <> ^ s1 = s0 ^
<y2> is prefix of s1 ^ s2 = s1[1,|s1|) ^ t2 = t1 ^
t3 = <y2> * t2 ^ s3 = s2 ^ T.Init(y3) → s3 /= <>

VC6: |s0| = 2 ^ T.Init(y1) ^ t1 = <> ^ s1 = s0 ^
<y2> is prefix of s1 ^ s2 = s1[1,|s1|) ^ t2 = t1 ^
t3 = <y2> * t2 ^ s3 = s2 ^ T.Init(y3) ^
<y4> is prefix of s3 ^ s4 = s3[1,|s3|) ^ t4 = t3 ^
t5 = <y4> * t4 ^ s5 = s4 ^ T.Init(y5) ^
s6 = t5 ^ t6 = <> ^ y6 = y5 → s6 = s0[1,2) * s0[0,1)

Steps:

1. Assume all the *premises* on the lhs of → are true
2. Use the premises from the lhs to show that the *conclusion* on the rhs cannot be false

- The proof goes through if the conclusion is shown to be true
- The proof fails if the conclusion cannot be shown to be true

To do: Prove VC6

Direct Proof of VC6

```
|s0| = 2 ^ T.Init(y1) ^ [6] t1 = <> ^ [7] s1 = s0 ^
<y2> is prefix of s1 ^ [9] s2 = s1[1,|s1|) ^ [5] t2 = t1 ^
[4] t3 = <y2> * t2 ^ [8] s3 = s2 ^ T.Init(y3) ^
<y4> is prefix of s3 ^ s4 = s3[1,|s3|) ^ [3] t4 = t3 ^
[2] t5 = <y4> * t4 ^ s5 = s4 ^ T.Init(y5) ^
[1] s6 = t5 ^ t6 = <> ^ y6 = y5 → s6 = s0[1,2) * s0[0,1)
```

Steps:
1) Assume all of VC6's premises are true
2) Show that the equality in VC6's conclusion holds utilizing its premises

Start with the VC6's conclusion (i.e., the rhs of VC6)
Do a backwards sweep starting with s6 using VC6's premises to make transformations to s6

1.	s6 = s0[1,2) * s0[0,1)	Conclusion from VC6
2.	t5 = s0[1,2) * s0[0,1)	Substitution using #1
3.	<y4> * t4 = s0[1,2) * s0[0,1)	Substitution using #2
4.	<y4> * t3 = s0[1,2) * s0[0,1)	Substitution using #3
5.	<y4> * <y2> * t2 = s0[1,2) * s0[0,1)	Substitution using #4
6.	<y4> * <y2> * t1 = s0[1,2) * s0[0,1)	Substitution using #5
7.	<y4> * <y2> * <> = s0[1,2) * s0[0,1)	Substitution using #6
8.	<y4> * <y2> = s0[1,2) * s0[0,1)	Concatenation identity
9.	s0[1,2) * s0[0,1) = s0[1,2) * s0[0,1)	Lemma #1 and Lemma #2 (below)

Lemma 1:

1.	<y2> is prefix of s1 = <y2> is prefix of s0	Substitution using #7
2.	<y2> = s0[0,1)	Definition of prefix

Lemma 2:

1.	<y4> is prefix of s3 = <y4> is prefix of s2	Substitution using #8
2.	<y4> = s2[0,1)	Definition of prefix
3.	<y4> = s1[1,|s1|)[0,1)	Substitution using #9
4.	<y4> = s0[1,|s0|)[0,1)	Substitution using #7
5.	<y4> = s0[1,2)	Application of Concise Notation