Loops in Assembly Language

Computer Science 111 Boston University

Vahid Azadeh-Ranjbar, Ph.D.

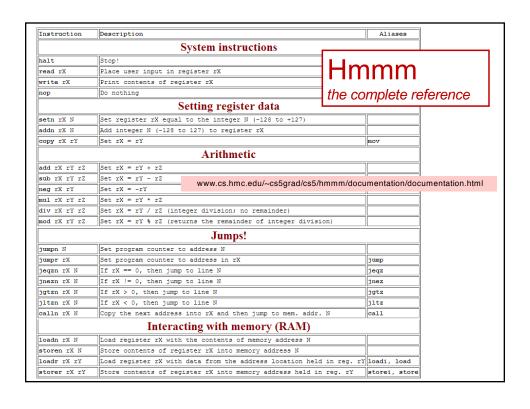
based in part on notes from the CS-for-All curriculum developed at Harvey Mudd College

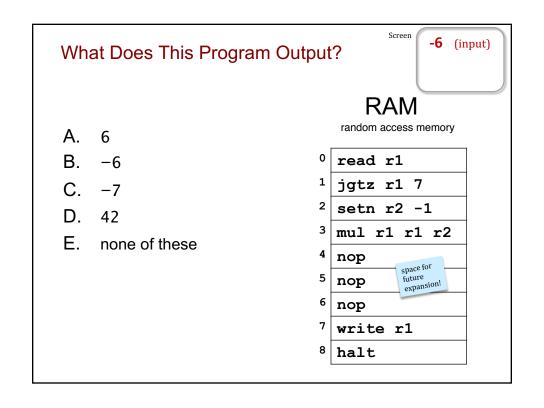
Jumps in Hmmm

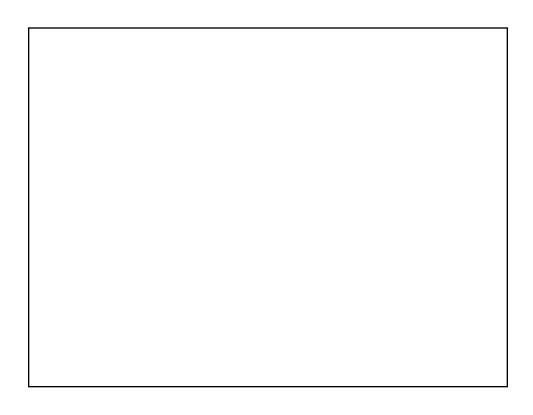
<u>instruction</u>	what it does	<u>example</u>
jeqz rX L	jumps to line L if rX == 0	jeqz r1 12
jgtz rX L	jumps to line L if rX > 0	jgtz r2 4
jltz rX L	jumps to line L if rX < 0	jltz r3 15
jnez rX L	jumps to line L if rX != 0	jnez r1 7
jumpn L	jumps to line L	jumpn 6
jumpr rX	jumps to line # stored in rX	jumpr r2
(more on this later)		

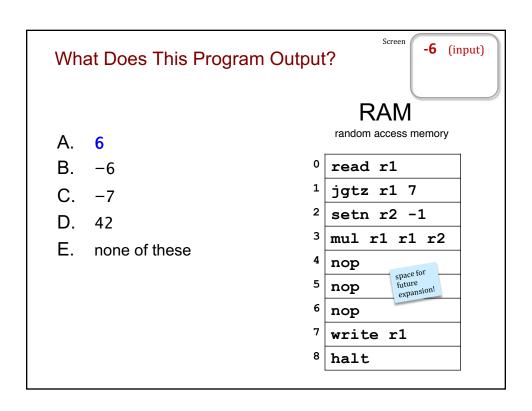
Notation:

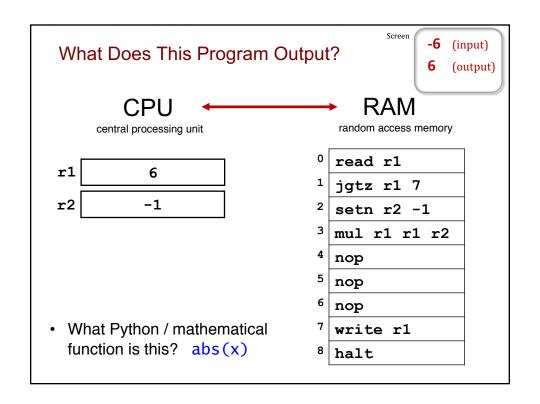
- rX is any register name (r1-r15)
- · L is the line number of an instruction

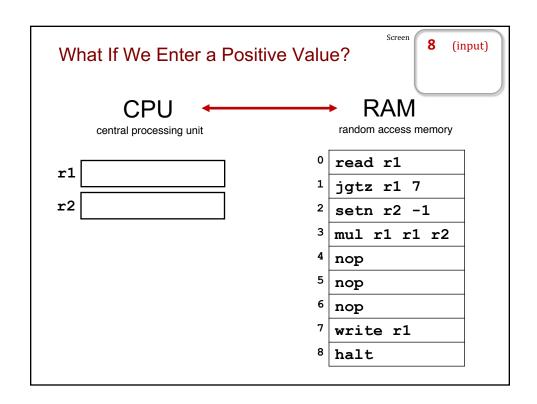


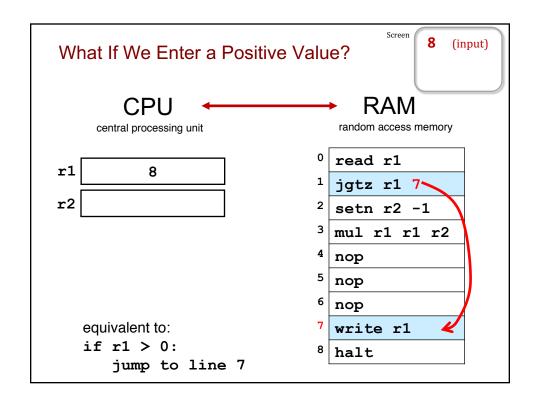


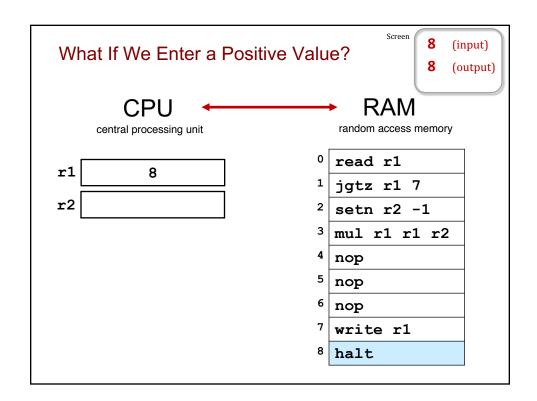












For the Inputs At Right, Which Lines Execute After the nop?

20 (1st input) **7** (2nd input)

- A. 4,5,6,8
- B. 4,7,8
- C. 4,5,6,7,8
- D. 4,6,8
- E. none of these
- What if we swap the two inputs?
- What Python function is this?
- read r1
 read r2
 sub r3 r1 r2
 nop
 jgtz r3 7
 write r1
 jumpn 8
 write r2
 halt
- How could you change <u>only line 3</u> so that, if the two inputs are equal, the program will ask for new inputs?

For the Inputs At Right, Which Lines Execute After the nop?

20 (1st input) **7** (2nd input)

- A. 4,5,6,8
- B. 4,7,8
- C. 4, 5, 6, 7, 8
- D. 4,6,8
- E. none of these
- · What if we swap the two inputs?
- What Python function is this?

0	read r1
1	read r2
2	sub r3 r1 r2
3	nop
4	jgtz r3 7
5	write r1
6	jumpn 8
7	write r2
8	halt

• How could you change <u>only line 3</u> so that, if the two inputs are equal, the program will ask for new inputs?

For the Inputs At Right, Which Lines Execute After the nop?

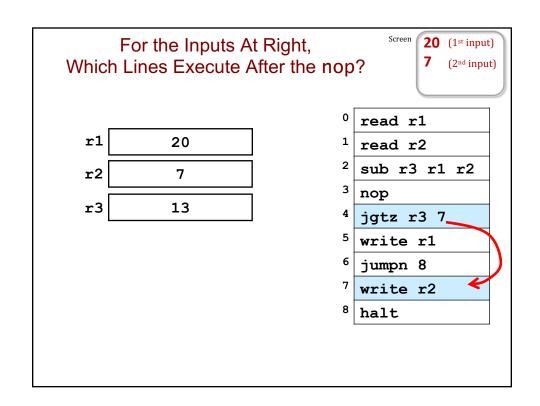
Screen **20** (1st input) **7** (2nd input)

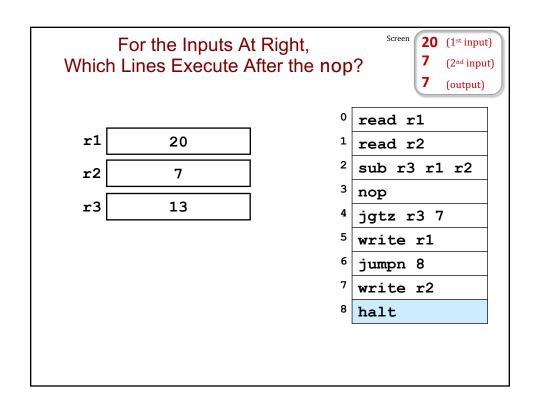
r1	20
r2	7
r3	13

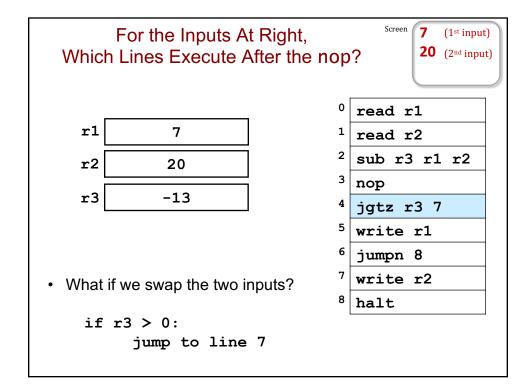
line 4 means:
if r3 > 0:
 jump to line 7

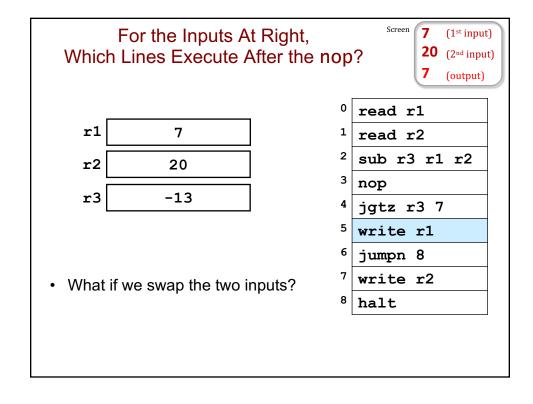
taken together,
lines 2 and 4 are equivalent to:
if r1 > r2:
 jump to line 7

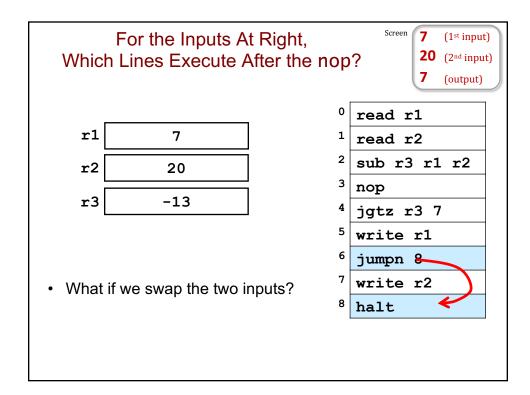
0	read r1
1	read r2
2	sub r3 r1 r2
3	nop
4	jgtz r3 7
5	write r1
6	jumpn 8
7	write r2
8	halt

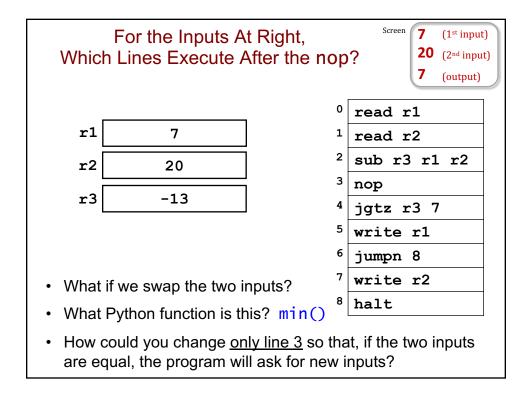


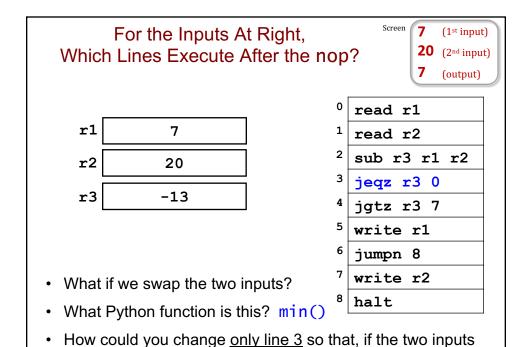




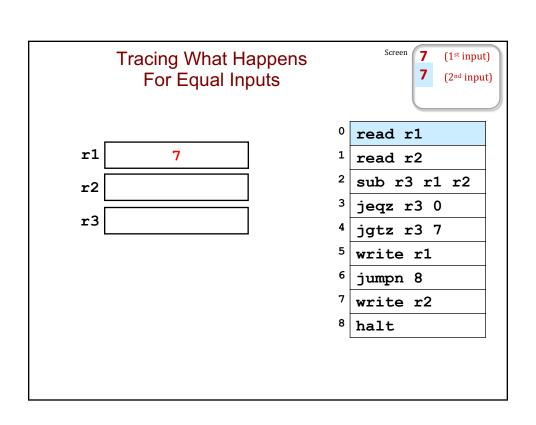


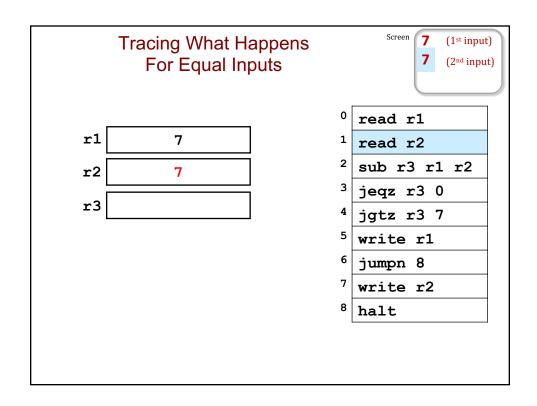


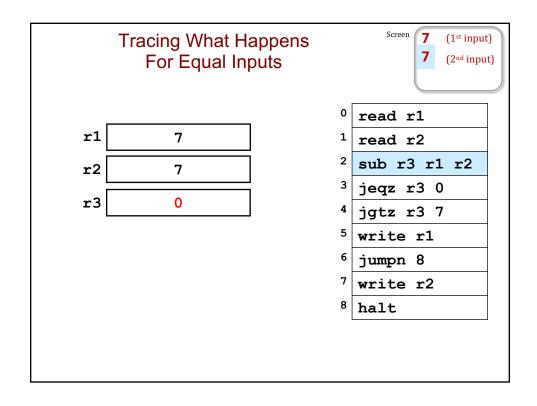


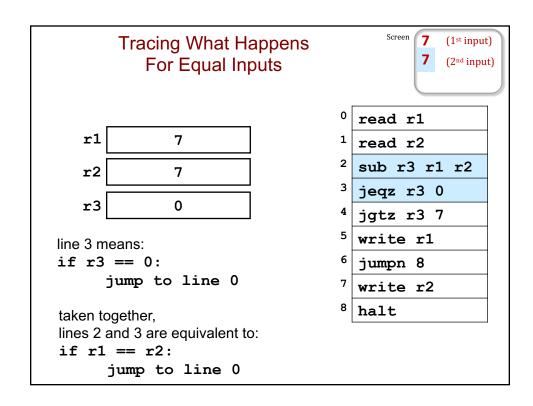


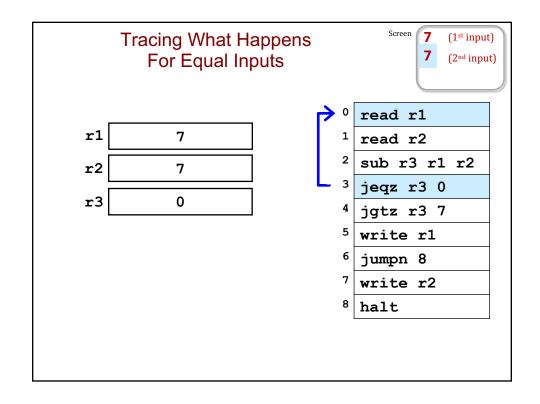
are equal, the program will ask for new inputs?

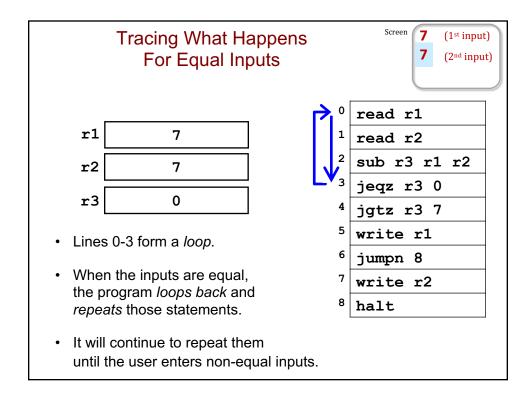


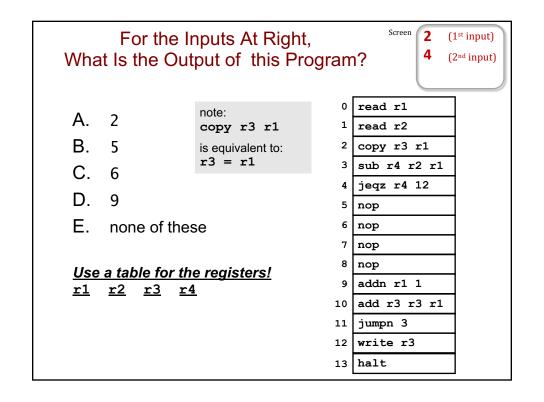


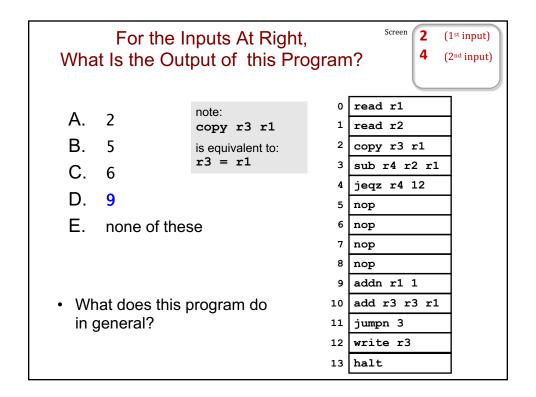


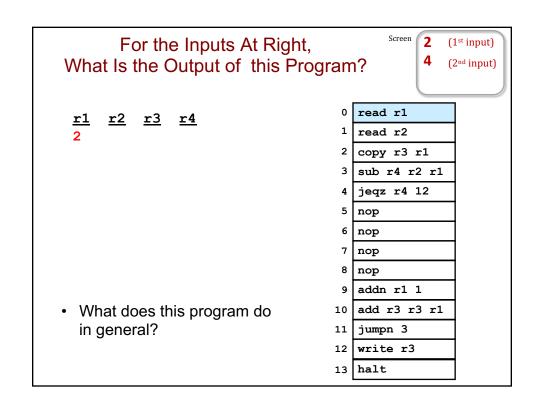


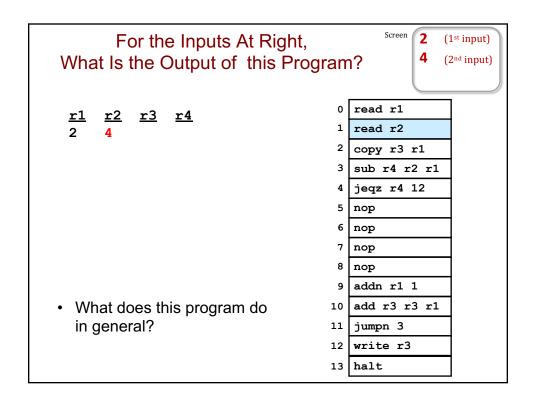


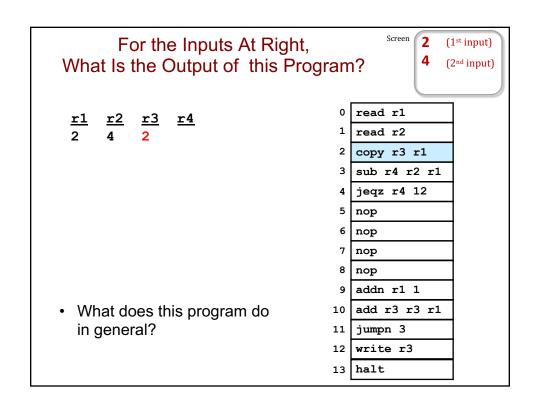


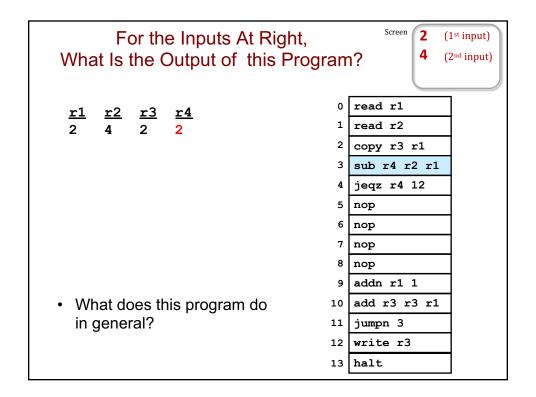


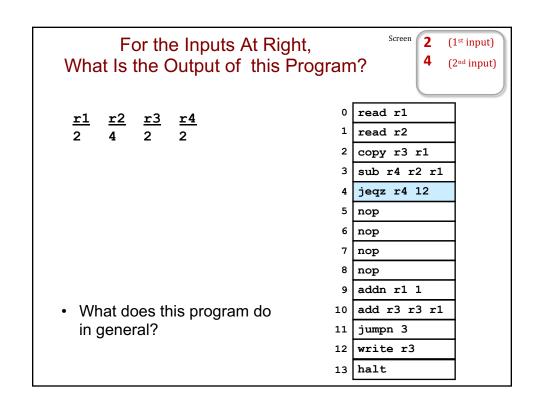


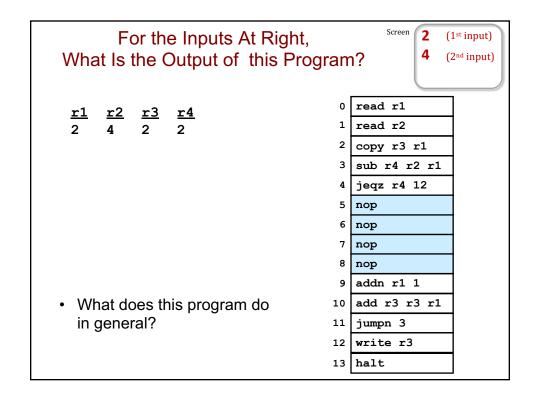


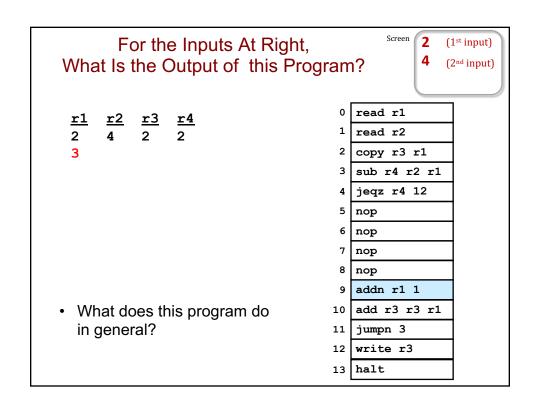


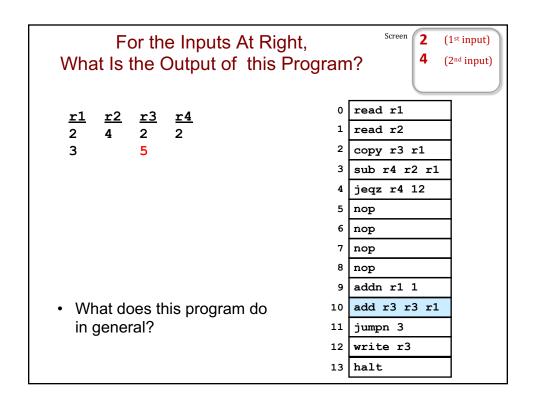


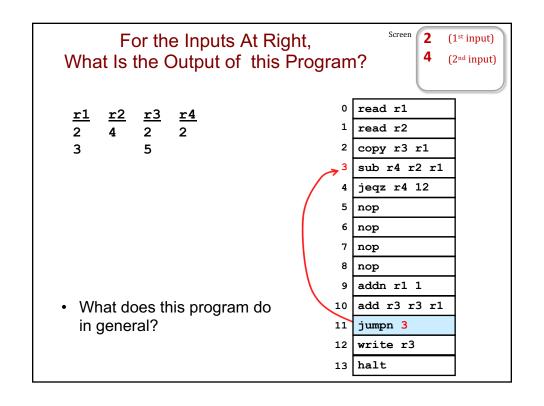


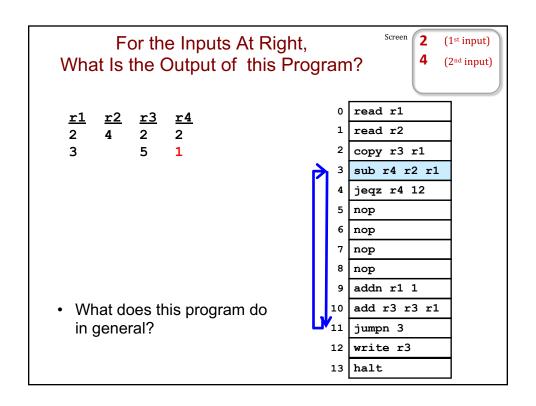


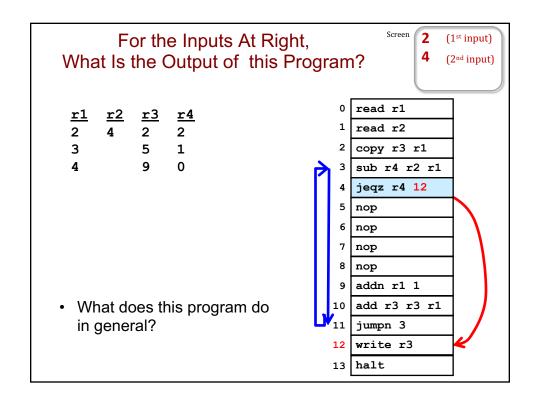


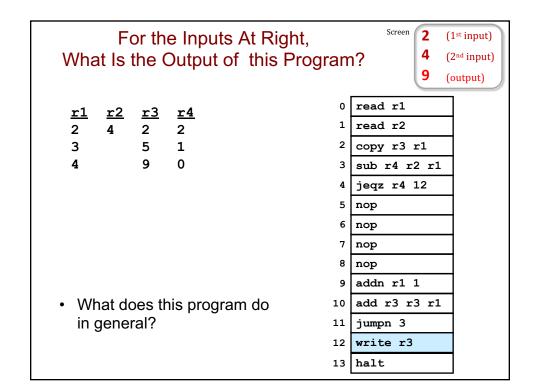


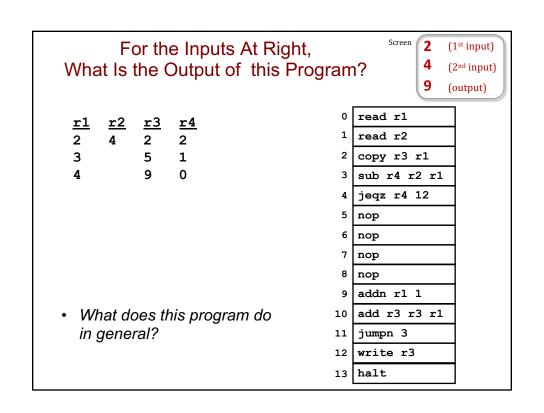












For the Inputs At Right, What Is the Output of this Program?

Screen 2 (1st input)
4 (2nd input)
9 (output)

<u>r1</u>	<u>r2</u>	<u>r3</u>	<u>r4</u>
2	4	2	2
3		5	1
4		9	0

0 read r1
1 read r2
2 copy r3 r1
3 sub r4 r2 r1
4 jeqz r4 12
5 nop
6 nop
7 nop
8 nop
9 addn r1 1
10 add r3 r3 r1
11 jumpn 3
12 write r3

 What does this program do in general? compute the sum the integers from the 1st input to the 2nd input

13 halt

Handling Invalid Inputs

Screen 4 (1st input)
2 (2nd input)

- What happens if the first input is larger than the second?
- Replace the nop instructions with ones that allow the program to still sum the integers in this case.
 - hint: you'll want to swap the values in r1 and r2, but only if necessary
 - you may need to change another register, too

read r1 read r2 copy r3 r1 sub r4 r2 r1 4 jegz r4 12 5 nop 6 nop 7 nop nop addn r1 1 add r3 r3 r1 jumpn 3 write r3 12 13 halt

Handling Invalid Inputs

- What happens if the first input is larger than the second? the program will keep looping
- Replace the nop instructions with ones that allow the program to still sum the integers in this case.
 - hint: you'll want to swap the values in r1 and r2, but only if necessary
 - you may need to change another register, too

Screen 4 (1st input)
2 (2nd input)

0	read r1
1	read r2
2	copy r3 r1
3	sub r4 r2 r1
4	jeqz r4 12
5	jgtz r4 9
6	copy r1 r2
7	copy r2 r3
8	copy r3 r1
9	addn r1 1
10	add r3 r3 r1
11	jumpn 3
12	write r3
13	halt