

What Would Python Do

Fill in the unfinished environment diagrams to match each block of code.

1. Stories

```
def quest(sword, robot):
    ogre = print(sword)
    sword, ogre = ogre, sword+robot
    robot /= 2
    return str(robot) > str(ogre)

robot = min(8, 8.0)
sword = int('5'+str(robot))
hero = quest(robot, sword)
```

Since 8 and 8.0 are equal, the min is just whichever comes first.

global	quest	→	function quest(sword, robot)
			[p=global]
	robot		8
	sword		58
	hero		False

f1: quest	sword	8	None
[p=global]	robot	58	29.0
	ogre	None	66
	return	False	

printed	8
output	

2. Painting

```
def paint(color):
    print(color)
    return color + str(print(color))
paint('Blue')
def print(paper):
    return max(paper * 2, 'Purple')
paint('Red')
```

At first, the variable `print` is bound to the built-in function that we talked about in the chapter. At this line, we override the built-in function by reassigning `print` as a pointer to a function of our own.

global	paint	→	function paint(color) [p=global]
	print	→	function print(paper) [p=global]

f1: paint	color	'Blue'
[p=global]	return	'BlueNone'

f2: paint	color	'Red'
[p=global]	return	'RedRedRed'

f3: print	paper	'Red'
[p=global]	return	'RedRed'

f4: print	paper	'Red'
[p=global]	return	'RedRed'

printed	Blue
output	Blue

3. Farm Business

```
tomato = 'pear'
def cost(fruit):
    return int(bool(fruit))
def pair(pear, fare):
    loss = max(cost(pear), fare)
    profit = bool(pear) * float(10 * fare // loss)
    return profit - loss
pear = 100 % pair(tomato, 3)
pumpkin = pair(pear, 4)
```

global	tomato	'pear'
	cost	→ function cost(fruit) [p=global]
	pair	→ function pair(pear, fare) [p=global]
	pear	2.0
	pumpkin	6.0

f1: pair [p=global]	pear	'pear'
	fare	3
	loss	3
	profit	10.0
	return	7.0

f2: cost [p=global]	fruit	'pear'
	return	1

f3: pair [p=global]	pear	2.0
	fare	4
	loss	4
	profit	10.0
	return	6.0

f4: cost [p=global]	fruit	2.0
	return	1

One Print To Rule Them All

Write what would be displayed from running the following line of code.

```
>>> print(print('61A', "is"), print(61), 'A')
```

61A is

61

None None A