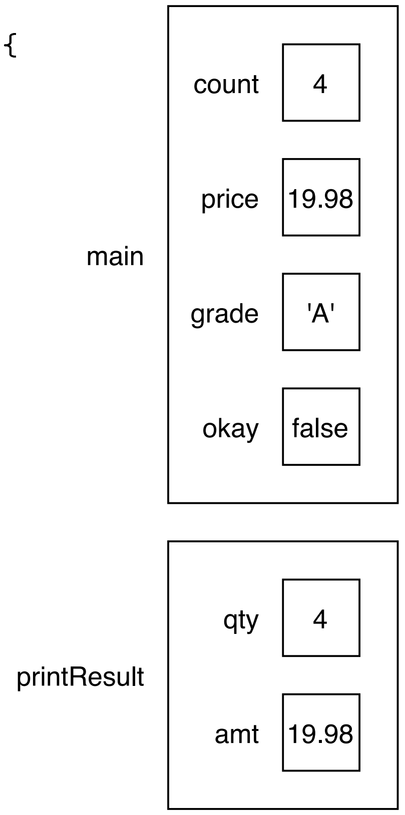
Local Variables

Consider the following example. The memory diagram shows the state of the program just before `printResult` returns for the second time:

```
public static void printResult(int qty, double amt) {
    System.out.printf("%d for $%.2f\n", qty, amt);
}

public static void main(String[] args) {
    int count = 3;
    double price = 9.99;
    char grade = 'A';
    boolean okay = true;
    printResult(count, price);
    count++;
    price *= 2;
    okay = !okay;
    printResult(count, price);
}
```



The output of the program is:

```
3 for $9.99
4 for $19.98
```

Questions (15 min)

Start time:

1. How many variables are declared ...

a) in `main`? 4

b) in `printResult`? 2

2. How many times is each variable assigned?

a) `count` 2 including ++

b) `price` 2

c) `grade` 1

d) `okay` 2

e) `qty` 2 two method calls

f) `amt` 2

3. Is there a small box for each declaration or each assignment? Justify your answer.

Each declaration; otherwise there would be more than four boxes in `main`.

4. What do the six small boxes in the memory diagram represent?

The contents of memory for each of the variables.

5. What do the two large boxes in the memory diagram represent?

The stack frames for each method, showing which variables are defined.

6. Why does the diagram indicate that `count` is `4` and `price` is `19.98`, even though the source code says that `count = 3` and `price = 9.99`?

The variables were modified later in the program. The diagram shows the state of memory near the end.

7. Based on the source code:

a) Which method is defined first? printResult

b) Which method is executed first? main

8. Copy the contents of *LocalVariables.java* into Java Visualizer. Click the "Visualize execution" button, and then click "Forward >" multiple times to see the code run.

a) What does the diagram look like on Step 11 of 19, just before `count++` executes?

There is only one frame (for `main`) with four variables: count=3, price=9.99, grade=`'A'`, and okay=true.

b) Why is there no frame for the `printResult` method on Step 11 of 19?

The method is not currently active; it returned during the previous step.

c) Run the program to Step 17 of 19, just before `printResult` returns for the second time. What differences do you notice between the diagram on the previous page and the one on Java Visualizer?

Answers might include:

- All boxes (variables and frames) have four sides in the activity.
- The frames are drawn in opposite order (top-down vs bottom-up).
- Java Visualizer labels the frames on top and shows the line number.
- Java Visualizer shows the method return values (even when `void`).