Memory Diagrams

When tracing code by hand, it's helpful to draw a picture to keep track of variables, methods, and objects. Memory diagrams represent the state of a program at a particular moment in time.

Manager:		Recorder:	
Presenter:		Reflector:	

Content Learning Objectives

After completing this activity, students should be able to:

- Describe primitive values and references in a memory diagram.
- Draw memory diagrams that have variables, arrays and objects.
- Summarize differences between variables, arrays, and objects.

Process Skill Goals

During the activity, students should make progress toward:

- Leveraging prior knowledge and experience of other students. (Teamwork)

Facilitation Notes

Discussion:
* Variables – all boxes are the "same" size (8 bytes or less)
* Pass int to a method, and it doesn't get changed in main
* Pass array to a method, and it gets changed (same object)

Misconceptions:
* Cannot have a reference to a variable (on the stack or heap)
* Assigning variable does not create a new box (only declaring)
* Assigning reference does not create a new object (only "new")

Note on strings:
* Shorthand – for convenience, objects may be drawn toString

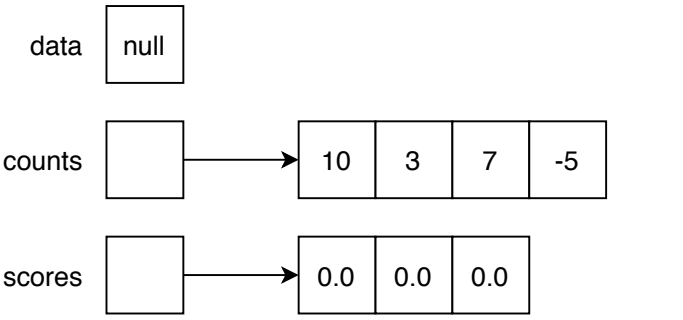
Key questions: #4, #11, #17

Source files: *Arrays.java*, *Card.java*, *BankAccount.java*

Model 1 Arrays

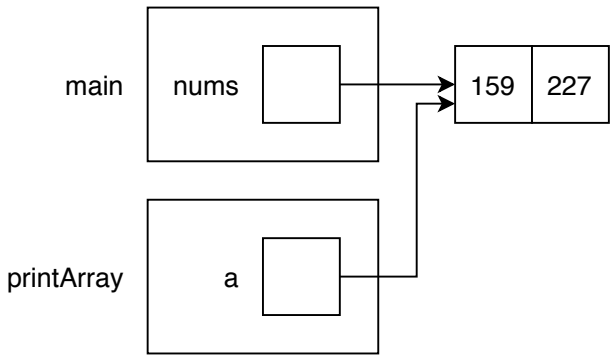
An array variable stores a ***reference*** to an array object. We draw references as arrows, because they "point" to other memory locations.

```
int[] data = null;

int[] counts = {10, 3, 7, -5};

double[] scores = new double[3];
```



When passing an array to a method, only the reference is copied:

```
public static void printArray(int[] a) {
    System.out.print("{" + a[0]);
    for (int i = 1; i < a.length; i++) {
        System.out.print(", " + a[i]);
    }
    System.out.println("}");
}

public static void main(String[] args) {
    int[] nums = {159, 227};
    printArray(nums);
}
```



Questions (15 min) Start time:

1. What is the length of each array?

a) `counts`? 4

b) `scores`? 3

c) `nums`? 2

d) `a`? 2

2. Looking at both diagrams above:

a) How many array variables were declared? 5

b) How many array objects were created? 3

3. Based on the top diagram, what is different about the variable named `data`?

The variable is `null`, so there is no reference and no array object.

4. Based on `counts` and `scores`, describe two ways that array objects can be created. How are these two ways different from each other?

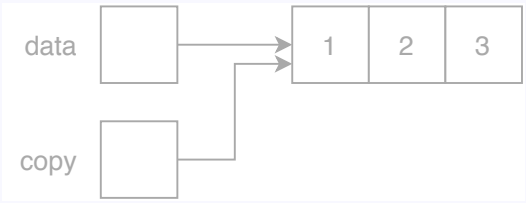
Arrays can be created either with curly braces or with the `new` keyword. The curly braces specify the initial values of array elements. The `new` keyword creates an array of specified length and initializes its elements to zero.

5. If the `printArray` method were to modify the array contents, would that change be visible in the `main` method? Explain your reasoning.

Yes, because the variable `nums` and the variable `a` reference the same object.

6. Draw (or describe) a diagram of the following source code:

```
int[] data = {1, 2, 3};
int[] copy = data;
```



7. (Optional) Paste the contents of *Arrays.java* into Java Visualizer. What differences do you notice between the diagram in Java Visualizer and those in Model 1?

Answers might include:

- The variables are drawn in (method) frames.
- The array objects show the index of each cell.

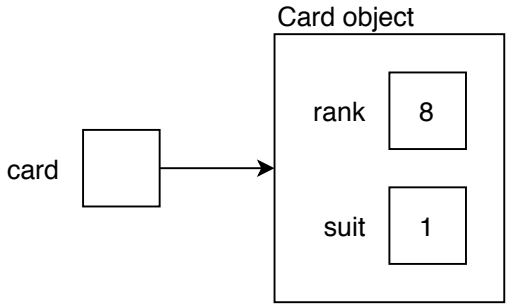
Model 2 Objects

Consider the definition for a playing card:

```
public class Card {
    private int rank;  // 1=Ace, ..., 11=Jack, 12=Queen, 13=King
    private int suit;  // 0=Clubs, 1=Diamonds, 2=Hearts, 3=Spades

    public Card(int rank, int suit) {
        this.rank = rank;
        this.suit = suit;
    }
}
```

Here is a memory diagram of a `Card` object:

```
Card card = new Card(8, 1);
```



Questions (15 min) Start time:

8. Which card (i.e., "the _____of _____") is represented in the diagram?

The 8 of Diamonds.

9. In one line of code, show how you would construct the "4 of Clubs".

```
Card card = new Card(4, 0);
```

10. What is the difference between lowercase `card` and uppercase `Card`? Explain in a few sentences how these concepts are illustrated in the diagram.

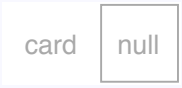
Lowercase `card` is a variable that contains a reference. Uppercase `Card` is a class, from which we create objects. Variables are small boxes; objects are large boxes that contain variables.

11. How are arrays and objects similar? How are arrays and objects different? Explain your answer in terms of how they are drawn in memory diagrams.

Both are reference types; their variables have an arrow pointing somewhere else. Arrays are drawn as contiguous boxes, representing multiple values of the same type. Objects are drawn as larger boxes, representing multiple variables by name.

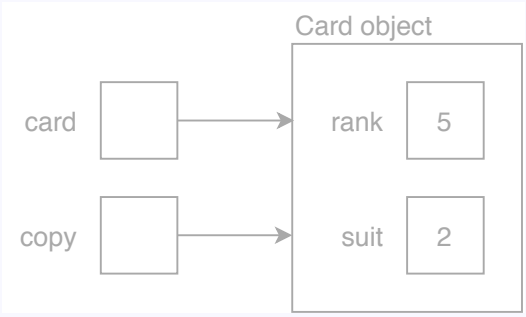
12. Draw (or describe) a diagram of the following source code:

```
Card card = null;
```



13. Draw (or describe) a diagram of the following source code:

```
Card card = new Card(5, 2);
Card copy = card;
```



14. (Optional) Paste the contents of *Card.java* into Java Visualizer. What differences do you notice between the diagram in Java Visualizer and those in Model 2?

Answers might include:

- JV says "Card instance" instead of "Card object".
- The object looks like a table, not separate boxes.

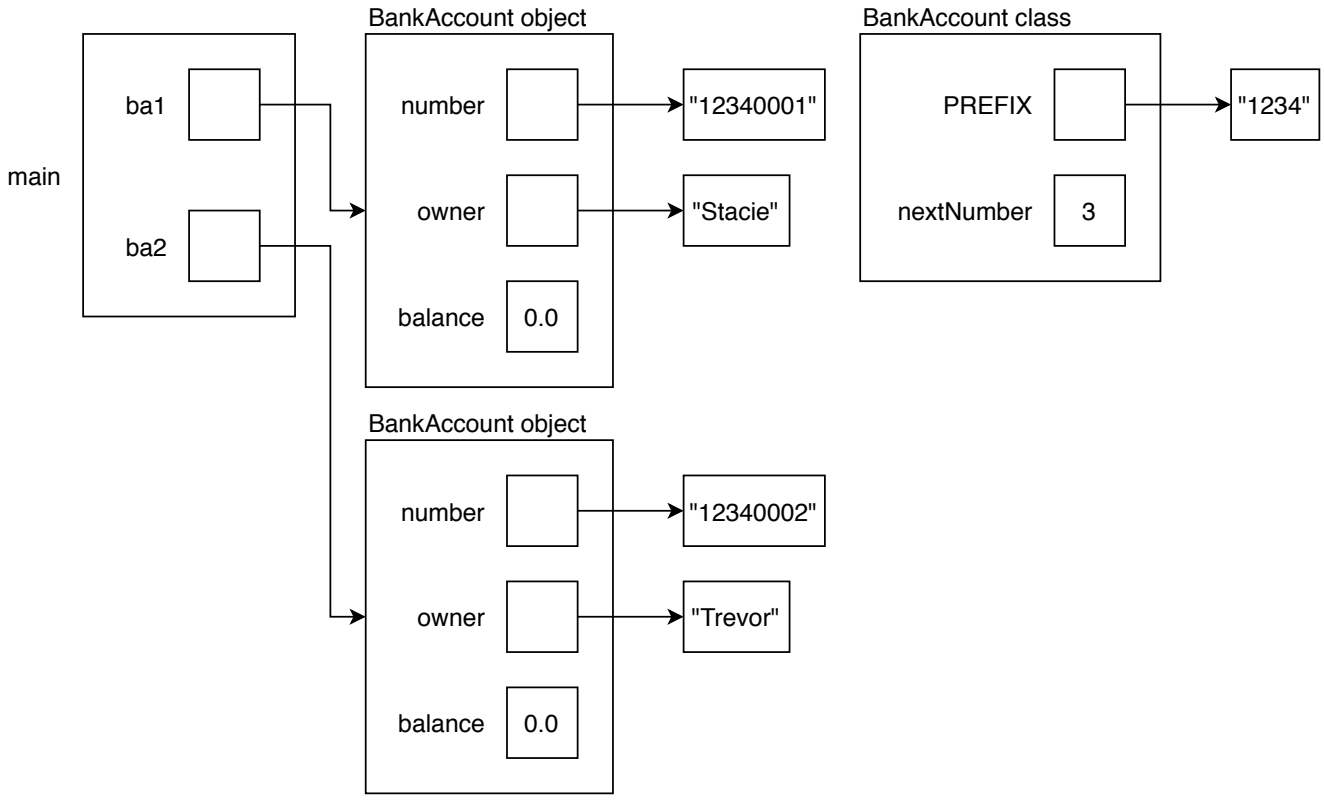
Model 3 Static Variables

Consider the definition for a bank account:

```
public class BankAccount {
    private static final String PREFIX = "1234";
    private static int nextNumber = 1;

    private String number;
    private String owner;
    private double balance;

    public BankAccount(String owner) {
        this.number = PREFIX + String.format("%04d", nextNumber);
        this.owner = owner;
        nextNumber++;
    }
}
```

Here is a memory diagram of two `BankAccount` objects:

```
public static void main(String[] args) {
    BankAccount ba1 = new BankAccount("Stacie");
    BankAccount ba2 = new BankAccount("Trevor");
}
```



Questions (15 min) **Start time:**

15. Based on the source code and memory diagram:

a) How many `BankAccount` variables were declared? 2

b) How many `BankAccount` objects were created? 2

16. How many instances of each variable are in memory?

a) `PREFIX` 1

b) `nextNumber` 1

c) `number` 2

d) `owner` 2

e) `balance` 2

17. What is the difference between `static` and non-`static` variables of a class? Explain your answer in terms of the diagram.

Static variables are shared by all instances of the class. They are shown in the box labeled "BankAccount class". Non-static variables are specific to each object. They are shown in boxes labeled "BankAccount object".

18. Why are all the strings shown in separate boxes as opposed to being written inside of the variable boxes?

Strings are objects, so their variables contain references. (There is not enough memory inside the variable to store the entire string.)

19. How would you modify the memory diagram if the following line were added at the end of the `main` method?

```
BankAccount ba3 = ba2;
```

Add a new box for `ba3` and draw an arrow to the object for `ba2`. Notice that assigning reference types does not create new objects!

20. (Optional) Paste the contents of *BankAccount.java* into Java Visualizer. What differences do you notice between the diagram in Java Visualizer and those in Model 3?

Answers might include:

- The static fields are drawn above the top frame.
- The strings are drawn inside the objects (for convenience).