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)

Model 1 Arrays

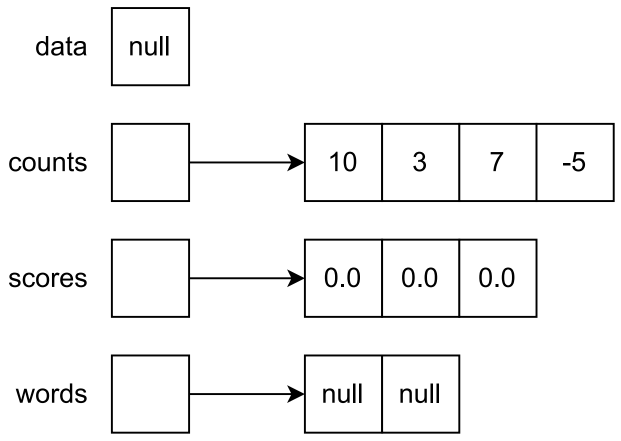
An array variable stores a ***reference*** to an array object. We draw references as arrows, because they “point” to other memory locations.

```
int[] data = null;

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

double[] scores = new double[3];

String[] words = new String[2];
```



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

1. What is the length of each array?

a) `counts`? b) `scores`? c) `words`?

2. Looking at the diagram above:

a) How many array variables were declared?

b) How many array objects were created?

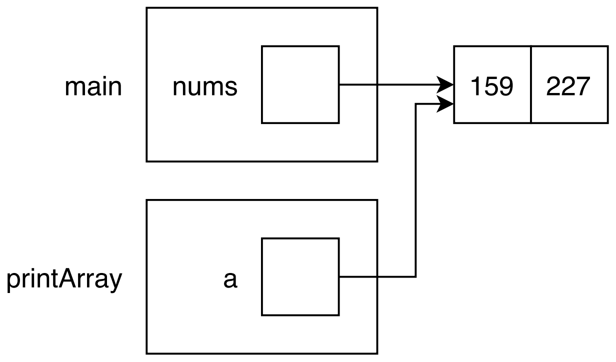
3. What is different about the variable named `data`?

4. Based on the code above, describe two ways that array objects can be created. How are these two ways different from each other?

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

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

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("}");
}
```



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

6. Draw (or describe) a diagram of the following code, and explain what it means to assign one array variable to another.

```
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?

Model 2 Objects

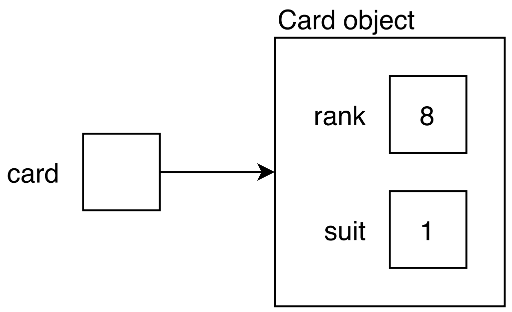
Consider the definition for a playing card:

```java
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:

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



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

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

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

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

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.

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?

Model 3 Static Variables

Consider the definition for a bank account:

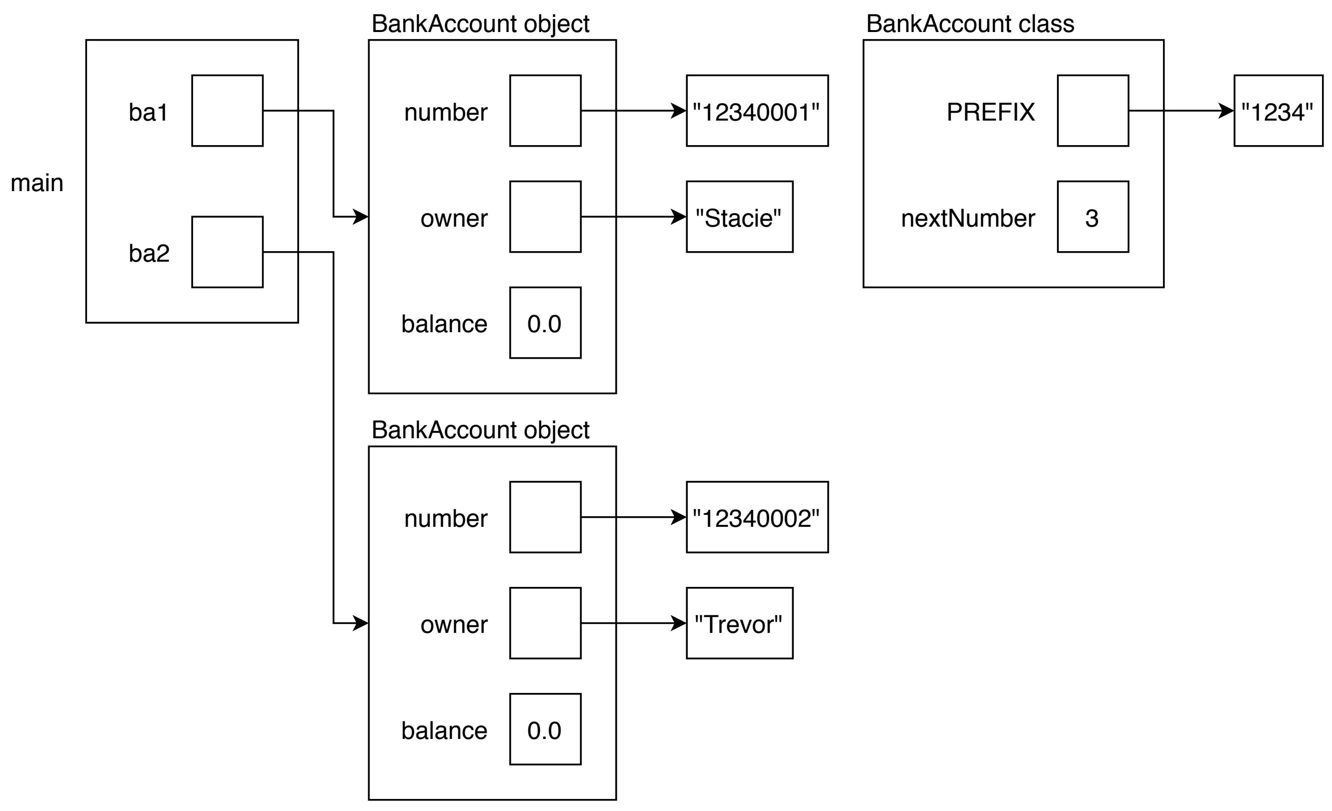
```java
public class BankAccount {
    private static final String PREFIX = "1234";
    private static int nextNumber = 1;

    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:

```java
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 variables of type `BankAccount` were declared?

b) How many objects of type `BankAccount` were created?

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

a) `PREFIX`

b) `nextNumber`

c) `number`

d) `owner`

e) `balance`

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

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

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

```
BankAccount ba3 = ba2;
```

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?