Arrays of Objects

With arrays and objects, you can represent pretty much any type of data. It's not only possible to have arrays of objects, but also objects of arrays, objects of objects, arrays of arrays, arrays of objects of arrays, and so forth.

Manager: **Alex Wang**

Recorder:

Presenter:

Reflector:

Content Learning Objectives

After completing this activity, students should be able to:

- Explain the difference of instantiating an array and an object.
- Rewrite a for loop (over an array) using an enhanced for loop.
- Use enhanced for loops to construct and search arrays of objects.

Process Skill Goals

During the activity, students should make progress toward:

- Developing algorithms for constructing and searching arrays. (Problem Solving)


Copyright © 2021 Chris Mayfield. This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.

Model 1 Hand of Cards

Creating an array of objects is typically a 3-step process:

1. Declare the array

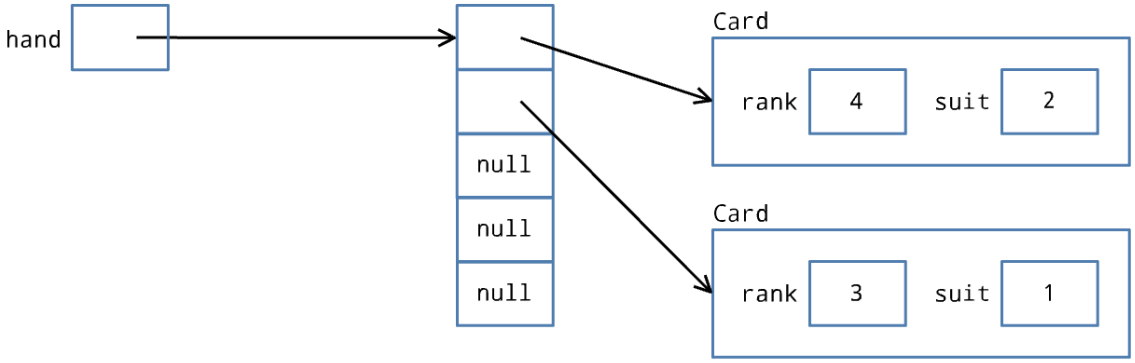
```
Card[] hand;
```

2. Instantiate the array

```
hand = new Card[5];
```

3. Instantiate each object

```
hand[O] = new Card(4, 2);
hand[1] = new Card(3, 1);
```



Questions (20 min) Start time:

1. What is the type of the local variable hand? What is the value of hand *before* step 2? What is the value of hand *after* step 2?

- **Type of hand is an array of Card instances**
- **Before step 2 it is just uninitialized and doesn't have a value**
- **After step 2 it is the memory location of the array**

2. When you create an array (e.g., new Card[5]) what is the initial value of each element?

Null for refs; 0 for int, 0.0 for double, false for bool, Unicode zero character for char

3. When you construct a new object (e.g., new Card(4, 2)) what are the initial values of its attributes (e.g., this.rank)?

This.rank is the first argument passed into the Card constructor; suit is the second argument

> *The new operator finds a memory location to store an array or object. Java automatically determines how much memory is needed and initializes the contents of the corresponding memory cells to zero. That's why array elements and object attributes have default values, whereas local variables (not allocated with new) must be initialized before they are used.*

4. Describe in your own words what the following code does. Be sure to explain how the random part

works.

```
int index = (int) (Math.random() * hand.length);
hand[index] = null;
```

The random part randomly selects an integer less or equal to than hand.length. This code randomly sets one of the Card references in hand to null.

5. What is the result of running the loop below? Explain why the if-statement is necessary.

```
for (int i = O; i < hand.length; i++) {
    if (hand[i] != null) {
        int suit = hand[i].getSuit();
        System.out.println("The suit of #" + i + " is " + Card.SUITS[suit]);
    }
}
```

An assignment error, because the value of int i cannot be initialized to the value "O".

6. The ***enhanced for loop*** allows you to iterate the elements of an array. Another name for this structure is the "for each" loop. Rewrite the following example using a standard for loop.

```
String[] days = {"Sun", "Mon", "Tue", "Wed", "Thu", "Fri", "Sat"};

for (String day : days) {
    System.out.println(day + "is a great day!");
}
```

for (int i = 0; i < days.length; i++) {
System.out.println(days[i] + " is a great day!");
}

7. In contrast to enhanced for loops, what does a standard for loop iterate? Why would it be misleading to name the enhanced for loop variable i instead of day?

The standard for loop iterates over the *indices* of an array. "i" is typically understood to refer to an index, so it would not make sense to name the enhanced for loop variable that (since the value of such a variable is not an array index).

8. Rewrite the loop in #5 using an enhanced for loop. Use an appropriate variable name for the Card object (i.e., not i). For simplicity, you may omit the System.out.println line.

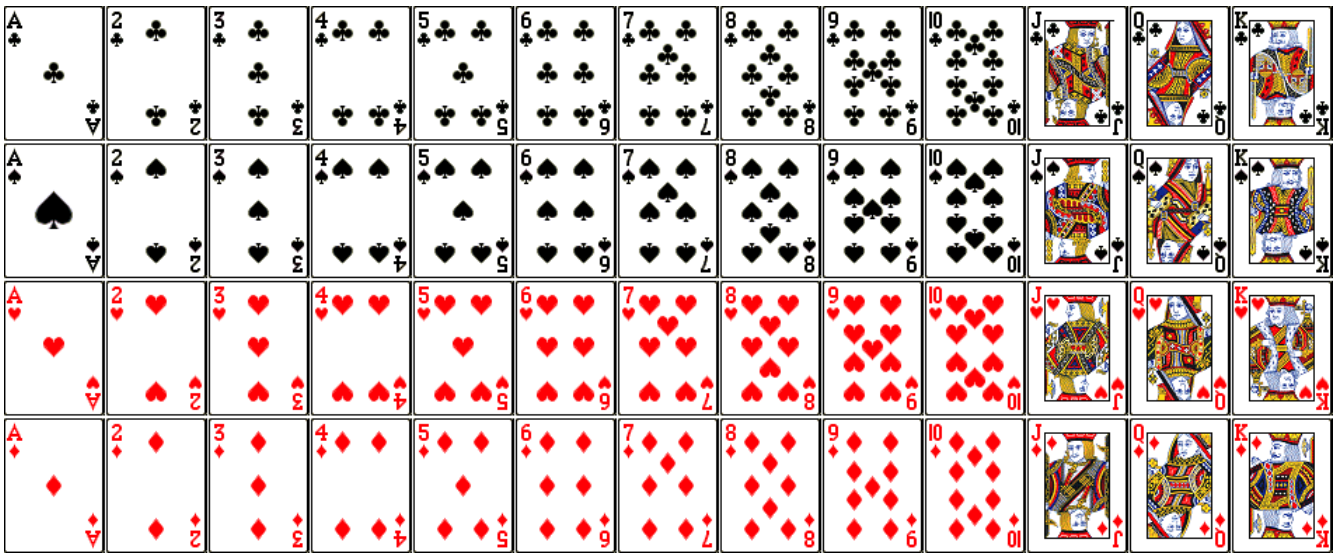
for (Card card : hand) {

```
if (card != null) {
int suit = card.getSuit();
}
}
```

Model 2 Deck of Cards

There are 52 cards in a standard deck. Each card has one of ***13 ranks*** (1=Ace, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11=Jack, 12=Queen, and 13=King) and one of ***4 suits*** (0=Clubs, 3=Spades, 2=Hearts, and 1=Diamonds). For example, new Card(12, 2) would construct the Queen of Hearts.

The following deck is represented by an array of Card objects. The array is one-dimensional, but the cards are shown in four rows (because of the paper margins).



Questions (25 min) Start time:

9. What is the index (in the array above) of the following cards?

a) Ace of Clubs **0**

b) Jack of Clubs **10**

c) 2 of Spades **14**

d) Queen of Spades **24**

e) 7 of Hearts **32**

f) King of Diamonds **51**

10. Write the following statements using one line of code each.

a) Declare and initialize a Card array named deck that can hold 52 cards.

Card[] deck = new Card[52];

b) Construct the Ace of Clubs, and assign it as the first element in deck.

deck[0] = new Card(1, 0);

c) Construct the King of Diamonds, and assign it as the last element in deck.

deck[deck.length - 1] = new Card(13, 1);

11. Describe how you could repeat code from the previous question to construct the entire deck of cards (without having to type 52 statements).

Use a nested for loop that iterates over card ranks and suits

12. Discuss the following code as a team:

```
int index = O;
int[] suits = {O, 3, 2, 1};
for (int suit  suits) {
        for (int rank = 1; rank <= 13; rank++) {
                deck[index] = new Card(rank, suit); index++;
        }
}
```

a) What is the overall purpose of the code?

Constructs an entire deck of 52 cards (1 card of each suit-rank combination)

b) Why is the suits array not just {0, 1, 2, 3}? (See Model 2.)

We want to follow the order of suits that was defined in the original question.

c) Why does the code use an enhanced for loop for suit?

The suits are out of order, and it's a lot easier to handle that with an enhanced for loop.

d) Why does the code use a standard for loop for rank?

The ranks are in numerical order so a normal for loop is a convenient way to handle that.

e) What is the purpose of the index variable?

The index variable keeps track of which item in the deck array is being filled by a card.

13. Write a method named inDeck that takes a Card[] representing a deck of cards and a Card object representing a single card, and that returns true if the card is somewhere in the deck.

```
public static boolean inDeck(Card[] deck, Card card) {
for (Card c : deck) {
if (c.equals(card)) {
```

return true;

}

}

return false;

}

Describe what the following code does and how it works. (Note: You've come a long way this semester, to be able to understand this example!)

```
public static Card[] sort(Card[] deck) {
        if (deck == null) {
            System.err.println("Missing deck!");
            return null;
        }
    Card[] sorted = new Card[deck.length];
    for (Card card  deck) {
        int index = card.position();          // returns suit * 13 + rank - 1
        sorted[index] = card;
    }
    return sorted;
}
```

a) What is the overall purpose of the code?

It sorts an array of cards and returns a new array that is sorted.

b) What is the purpose of the if statement?

It makes sure the deck passed into the method has actually been initialized.

c) Does this method modify the deck array? Justify your answer.

No, "deck" is never assigned to anything in the method.

d) How does the sort method know where to put each card?

It's based on the .position() method built into the Card class.

14. Identify the following Java language features in the previous question.

a) Variables **deck, card, index, sorted**

b) Decisions **if (deck === null)**

c) Loops **for (Card card : deck)**

d) Methods **public static Card[] sort…**

e) Arrays **deck, sorted**

f) Objects **card, String "Missing deck!"**