Java 9/26-28

Java Fundamentals - Arrays

- one dimensional arrays
- multidimensional arrays
- copy array

is a group of like type of variables with like type of names

variable in an array are ordered, each has an index of 0

used as static field, a local variable, or method parameter

How to declare 1 dimensional array

dataType[] arrayName;

“one line of boxes of shoes”

like type

dataType: primitive data types

looping through elements in a array

we use the for loop to it

Copy Array

Challenge-

5 apps

5entrees

5 desserts

20 orders from drivetrue

only allowed to

three arrays …one for food, one for pricing, one for empty array

string, double, and int…starting at zero

Object Oriented Programming

an object is an entity that has state and behaviors

java class is a blueprint for the object

an object is called an instance of a class.

look over constructors and what it does

accessing members of a class

we can use the name of the objects along with the .operator to access the member of a class

two types of methods

user-defined methods

standard library methods

returnType - It specifies what type of value a method returns

if a method has an int return type then it returns an integer value.

If the method does not return a value, its return type is void.

why we use methods

code reuasblity

```
import java.util.InputMismatchException;
import java.util.Scanner;

 class Book{
   String nameOfBook;
   String authorFirstName;
```

```
    String authorLastName;
    String genre;
    String ISBN;
    String publishDate;
    String price;
  }

  class Library {

    String [] customerArray;
    String date;
    boolean inStock;

    Book [] Books;




    String GetBookTitle(String identifier) {
      for(Book book : Books){
        if(book.ISBN == identifier) {
          return book.nameOfBook;
        }
        //search through book array for identifier (ISBN)
      }
      return "Did not find book";
      //if can't find it, return "Can't find it"
      //else return the nameOfBook from Book class

    }
    void AddNewBook() {

    }
    void RemoveFromInventory() {

    }
    boolean OrderNewBook(){

    }
    void StockBook(){

    }
    String CheckoutMethod(){

    }


   public static void main(String[] args) {

     Library myLib = new Library();
     myLib.RemoveFromInventory();
   }
 }
```

Java Method Overloading

Constructors

1. no-arg constructor:
2. java parameterized constructor
3. default constructor

```java
import java.util.Random;

public class RestaurantMenu {
    public static void main(String[] args) {
        // Create arrays for appetizers, entrees, and desserts
        MenuItem[] appetizers = new MenuItem[5];
        MenuItem[] entrees = new MenuItem[5];
        MenuItem[] desserts = new MenuItem[5];

        // Populate the arrays with items
        // (The code for populating these arrays remains the same as in your previous code)

        // Combine all food items into a single array
        MenuItem[] allFoodItems = new MenuItem[15];
        System.arraycopy(appetizers, 0, allFoodItems, 0, appetizers.length);
        System.arraycopy(entrees, 0, allFoodItems, appetizers.length, entrees.length);
        System.arraycopy(desserts, 0, allFoodItems, appetizers.length + entrees.length, desserts.length);

        // Create a new array for 20 random food items
        MenuItem[] randomFoodItems = new MenuItem[20];

        // Generate random indices and add items to the new array
        Random random = new Random();
        for (int i = 0; i < 20; i++) {
            int randomIndex = random.nextInt(allFoodItems.length);
            randomFoodItems[i] = allFoodItems[randomIndex];
        }

        // Display the random food items
        System.out.println("Random Food Items:");
        for (int i = 0; i < randomFoodItems.length; i++) {
            System.out.println(randomFoodItems[i]);
        }
    }
}

class MenuItem {
    private String name;
    private double price;

    public MenuItem(String name, double price) {
        this.name = name;
        this.price = price;
    }
    @Override
    public String toString() {
        return name + ": $" + price;
    }
}
```