Assignment 2 Lab 5

Author: Aashay Pawar
NUID: 002134382

Problem 1 (50 points)
To help you understand wrapper classes, we want you to design your Integer-like wrapper class. Please design a class named MyInteger. The class contains:

- An int data field named value that stores the int value represented by this object.
- A constructor that creates a MyInteger object for the specified int value. • A getter method that returns the int value.
- The methods isEven(), isOdd(), and isPrime() that return true if the value in this object is even, odd, or prime, respectively.
- The static methods isEven(int), isOdd(int), and isPrime(int) that return true if the specified value is even, odd, or prime, respectively.
- The static methods isEven(MyInteger), isOdd(MyInteger), and isPrime(MyInteger) that return true if the specified value is even, odd, or prime, respectively.
- The methods equals(int) and equals(MyInteger) that return true if the value in this object is equal to the specified value.
- A static method parseInt(char[]) that converts an array of numeric characters to an int value.
- A static method parseInt(String) that converts a string into an int value.

Problem Description: The problem requires designing a class named MyInteger that serves as a wrapper for the primitive int data type. The class should provide functionality to store and manipulate int values, including determining if a value is even, odd, or prime, comparing values, and converting numeric characters or strings to int values.

Analysis: To design the MyInteger class, we need to include the following components:
1. Data Field:
   - `value` (int): Stores the int value represented by the MyInteger object.
2. Constructors:
   - `MyInteger(int value)`: Initializes a MyInteger object with the specified int value.
3. Getter Method:
   - `getValue()`: Returns the int value stored in the MyInteger object.
4. Instance Methods:
   - `isEven()`: Returns true if the value in the MyInteger object is even.
   - `isOdd()`: Returns true if the value in the MyInteger object is odd.
   - `isPrime()`: Returns true if the value in the MyInteger object is prime.
   - `equals(int value)`: Returns true if the value in the MyInteger object is equal to the specified value.
   - `equals(MyInteger myInt)`: Returns true if the value in the MyInteger object is equal to the value in the specified MyInteger object.
5. Static Methods:
   - `isEven(int value)`: Returns true if the specified value is even.
   - `isOdd(int value)`: Returns true if the specified value is odd.
   - `isPrime(int value)`: Returns true if the specified value is prime.
   - `isEven(MyInteger myInt)`: Returns true if the value in the specified MyInteger object is even.

- `isOdd(MyInteger myInt)`: Returns true if the value in the specified MyInteger object is odd.
- `isPrime(MyInteger myInt)`: Returns true if the value in the specified MyInteger object is prime.
- `parseInt(char[] numericChars)`: Converts an array of numeric characters to an int value.
- `parseInt(String numericString)`: Converts a string into an int value. With this analysis, we can proceed to implement the MyInteger class based on the provided requirements.

Solution:

```
package edu.northeastern.csye6200;


public class LAB5P1 {
	public static void main(String[] args) {
		// TODO: write your code here
		try {
			//create two objects of type MyInteger
			MyInteger m1 = new MyInteger(7);
			MyInteger m2 = new MyInteger(24);
			char[] charArr = { '4', '3', '7', '8'};
			System.out.println("n1 is even? " + m1.isEven());
			System.out.println("n1 is prime? " + m1.isPrime());
			System.out.println("15 is prime? " + MyInteger.isPrime(15));
			System.out.println("parseInt(char[]) for \"{ '4', '3', '7', '8' }\" = " +
MyInteger.parseInt(charArr));
			System.out.println("parseInt(String) for \"4378\" = " +
MyInteger.parseInt("4378"));
			System.out.println("n2 is odd? " + MyInteger.isOdd(45));
			System.out.println("45 is equal to n2? " + m1.equals(m2));
			System.out.println("n1 is equal to 5? " + m1.equals(5));
			System.out.println("n1 is odd? " + m2.isOdd());
		}
		catch(Exception e) {
```

```
                        System.out.print(e);
                }
        }
}

class MyInteger {

        // TODO: write your code here
        int mValue;

        public int getValue() {
                // TODO: write your code here
                return mValue;
        }

        public MyInteger(int value) {
                // TODO: write your code here
                this.mValue = value;
        }

        public boolean isPrime() {
                // TODO: write your code here
                return isPrime(this.mValue);
        }

        public static boolean isPrime(int num) {
                // TODO: write your code here
                if(num < 1 || num == 1) return false;
                for(int i = 2; i < num/2; i++) {
                        if(num % i == 0) return false;
```

```
        }
        return true;
    }

    public static boolean isPrime(MyInteger o) {
        // TODO: write your code here
        return isPrime(o.mValue);
    }

    public boolean isEven() {
        // TODO: write your code here
        return mValue % 2 == 0;
    }

    public boolean isOdd() {
        // TODO: write your code here
        return mValue % 2 != 0;
    }

    public static boolean isEven(int n) {
        // TODO: write your code here
        return n%2 == 0;
    }

    public static boolean isOdd(int n) {
        // TODO: write your code here
        return !isEven(n);
    }

    public static boolean isEven(MyInteger n) {
```

```
        // TODO: write your code here
        return n.isEven();
    }

    public boolean equals(int anotherNum) {
        // TODO: write your code here
        return this.mValue == anotherNum;
    }

    public boolean equals(MyInteger o) {
        // TODO: write your code here
        return this.mValue == o.mValue;
    }

    public static int parseInt(char[] numbers) {
        // numbers consists of digit characters.
        // For example, if numbers is {'1', '2', '5'}, the return value
        // should be 125. Please note that
        // numbers[0] is '1'
        // numbers[1] is '2'
        // numbers[2] is '5'

        // TODO: write your code here
        return Integer.parseInt(new String(numbers));

    }

    public static int parseInt(String s) {
        // s consists of digit characters.
        // For example, if s is "125", the return value
```

```
                // should be 125.


                // TODO: write your code here
                return Integer.parseInt(s);
        }
}
```

Output:

```
n1 is even? false
n1 is prime? true
15 is prime? false
parseInt(char[]) for "{ '4', '3', '7', '8' }" = 4378
parseInt(String) for "4378" = 4378
n2 is odd? true
45 is equal to n2? false
n1 is equal to 5? false
n1 is odd? false
```

Problem 2 (50 points)

We want you to create a class RoomPeople that can be used to record the number of people in the rooms of a building. The class has the attributes:

- numberInRoom - the number of people in a room
- totalNumber - the total number of people in all rooms as a static variable The class has the following methods:
- addOneToRoom - adds a person to the room and increases the value of totalNumber
- removeOneFromRoom - removes a person from the room, ensuring that numberInRoom does not go below zero, and decreases the value of totalNumber as needed
- getNumber - returns the number of people in the room
- getTotal - a static method that returns the total number of people Please write a program to test the class RoomPeople.

Problem Description: The problem requires creating a class named RoomPeople that can be used to record the number of people in the rooms of a building. The class should have attributes to store the number of people in a room and the total number of people in all rooms. It should also provide methods to add or remove people from a room, retrieve the number of people in a room, and retrieve the total number of people.

Analysis: To design the RoomPeople class, we need to include the following components:

1. Attributes:
   - `numberInRoom` (int): Represents the number of people in a room.

- `totalNumber` (static int): Represents the total number of people in all rooms.

2. Methods:
   - `addOneToRoom()`: Adds a person to the room by incrementing the value of `numberInRoom` and `totalNumber`.
   - `removeOneFromRoom()`: Removes a person from the room by decrementing the value of `numberInRoom` and `totalNumber`. Ensures that `numberInRoom` does not go below zero.
   - `getNumber()`: Returns the number of people in the room (`numberInRoom`).
   - `getTotal()`: Returns the total number of people across all rooms (`totalNumber`) as a static method. With this analysis, we can proceed to implement the RoomPeople class based on the provided requirements.

Solution:

```
package edu.northeastern.csye6200;


public class LAB5P2 {
	public static void main(String[] args){
		// TODO: write your code here
		RoomPeople a = new RoomPeople();
		RoomPeople b = new RoomPeople();
		System.out.println("Add two to room a and three to room b");
		//add two to room A
		a.addOneToRoom();
		a.addOneToRoom();
		//add three to room B
		b.addOneToRoom();
		b.addOneToRoom();
		b.addOneToRoom();
		System.out.println("Room a holds " + a.getNumber());
		System.out.println("Room b holds " + b.getNumber());
		System.out.println("Total in all rooms is " + RoomPeople.getTotal());
		System.out.println("Remove two from both rooms");
		a.removeOneFromRoom();
		a.removeOneFromRoom();
		b.removeOneFromRoom();
```

```
                b.removeOneFromRoom();
                System.out.println("Room a holds " + a.getNumber());
                System.out.println("Room b holds " + b.getNumber());
                System.out.println("Total in all rooms is " + RoomPeople.getTotal());
                System.out.println("Remove two from room a (should not change the
values)");
                a.removeOneFromRoom();
                a.removeOneFromRoom();
                System.out.println("Room a holds " + a.getNumber());
                System.out.println("Room b holds " + b.getNumber());
                System.out.println("Total in all rooms is " + RoomPeople.getTotal());
        }
}

class RoomPeople {

        // TODO: write your code here
        int n;
        static int total= 0;

   public static int getTotal(){
        // TODO: write your code here
        return total;
   }

   public RoomPeople() {
        // TODO: write your code here
        this.n = 0;
   }
```

```
    public void addOneToRoom(){
        // TODO: write your code here
        n = n + 1;
        total= total + 1;
    }

    public void removeOneFromRoom(){
        // TODO: write your code here
        if(this.n > 0) {
            this.n = this.n - 1;
            total = total - 1;
        }
    }

    public int getNumber(){
        // TODO: write your code here
        return n;
    }
}
```

Output:

```
Add two to room a and three to room b
Room a holds 2
Room b holds 3
Total in all rooms is 5
Remove two from both rooms
Room a holds 0
Room b holds 1
Total in all rooms is 1
Remove two from room a (should not change the values)
```

```
Room a holds 0
Room b holds 1
Total in all rooms is 1
```

Problem 3 (Optional for Extra Credit: 10 points)
In this problem, we want to design a shopping cart system. Specifically, you need to implement Cart and Product classes with the following requirements: Product class has the following attributes and methods:

- itemName - name of the item
- price - price of the item
- product(String itemName, double price) - a constructor with parameters itemName, price
- getItemName() - returns the itemName attribute
- getPrice() - returns the itemPrice attribute
- toString() – returns itemName and price as a String

Cart class has the following attributes and methods:

- products - list of all the items in the cart separated by comma
- cartTotal - total cart value
- A default (no-arg) constructor that creates a Cart object with default values (“” for products and 0.0 for cartTotal)
- getCartTotal() - returns the cartTotal attribute
- addProduct(Product product) - add the new item to the cart
- calculateChange(double payment) - returns the change amount after payment
- toString() – returns all available items in the Cart as a String, i.e., Cart { “xxxx”, “xxxx”, ... }

Solution:

```
package edu.northeastern.csye6200;


import java.util.List;
import java.util.ArrayList;


public class Cart {
    private List<Product> products;
    private double cartTotal;


    public Cart() {
        products = new ArrayList<>();
        cartTotal = 0.0;
```

```
    }

    public double getCartTotal() {
        return cartTotal;
    }

    public void addProduct(Product product) {
        products.add(product);
        cartTotal += product.getPrice();
    }

    public double calculateChange(double payment) {
        return payment - cartTotal;
    }

    @Override
    public String toString() {
        StringBuilder sb = new StringBuilder("Cart{");
        for (int i = 0; i < products.size(); i++) {
            sb.append(" \"").append(products.get(i).getItemName()).append("\"");
            if (i != products.size() - 1) {
                sb.append(",");
            }
        }
        sb.append(" }");
        return sb.toString();
    }
}
```

```
package edu.northeastern.csye6200;
```

```java
public class Product {
    private String itemName;
    private double price;

    public Product(String itemName, double price) {
        this.itemName = itemName;
        this.price = price;
    }

    public String getItemName() {
        return itemName;
    }

    public double getPrice() {
        return price;
    }

    @Override
    public String toString() {
        return "Product{ itemName=\"" + itemName + "\", price=$" + price + " }";
    }
}
```

```java
package edu.northeastern.csye6200;

public class LAB5P3 {
    public static void main(String[] args) {
        // Create products
        Product p1 = new Product("Milk", 3.7);
```

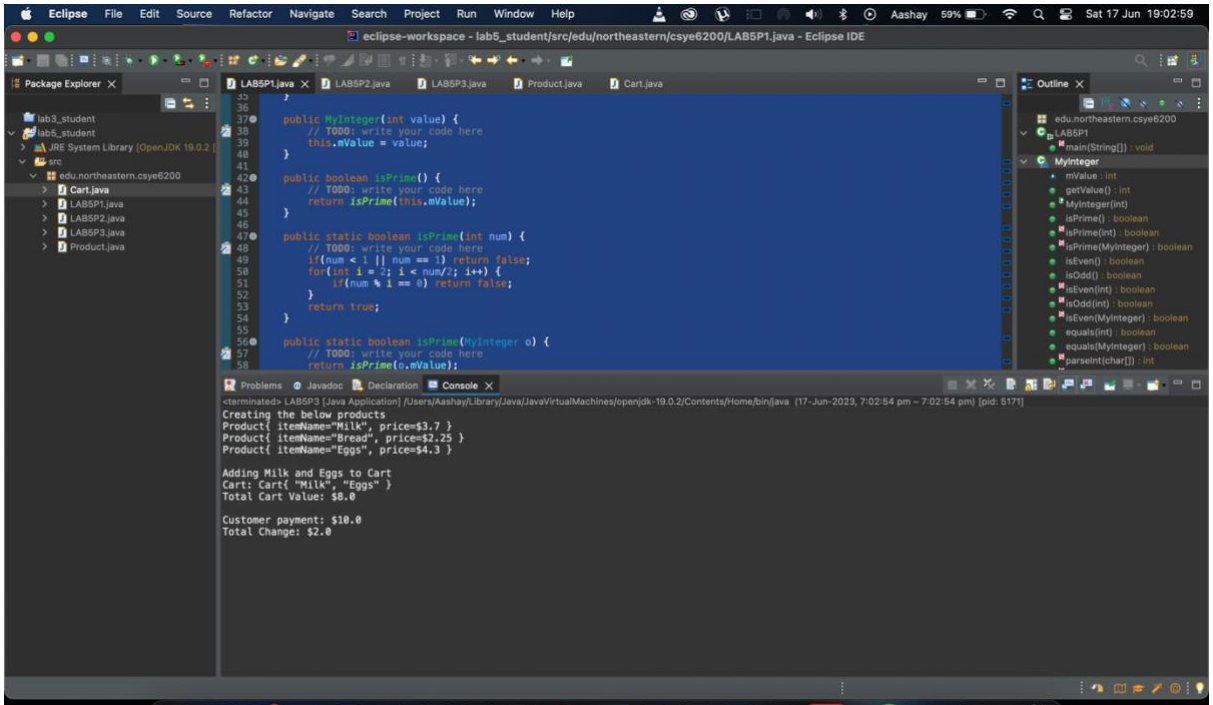
```
        Product p2 = new Product("Bread", 2.25);
        Product p3 = new Product("Eggs", 4.3);

        // Create cart
        Cart cart = new Cart();

        // Add products to the cart
        cart.addProduct(p1);
        cart.addProduct(p3);

        // Print the products
        System.out.println("Creating the below products");
        System.out.println(p1);
        System.out.println(p2);
        System.out.println(p3);

        // Print adding products to the cart
        System.out.println("\nAdding Milk and Eggs to Cart");

        // Print the cart
        System.out.println("Cart: " + cart);

        // Print the total cart value
        System.out.println("Total Cart Value: $" + cart.getCartTotal());

        // Calculate change after payment
        double payment = 10.0;
        double change = cart.calculateChange(payment);

        // Print the payment and change
```

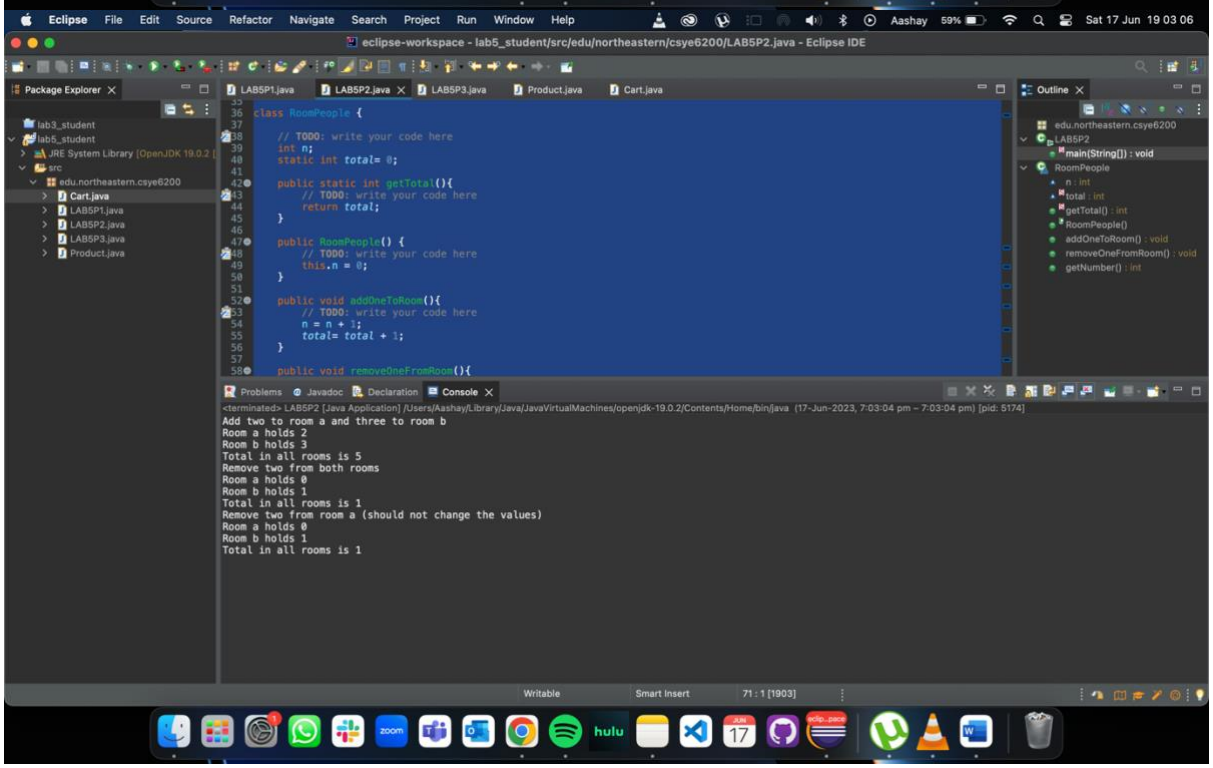
```
        System.out.println("\nCustomer payment: $" + payment);
        System.out.println("Total Change: $" + change);
    }
}
```

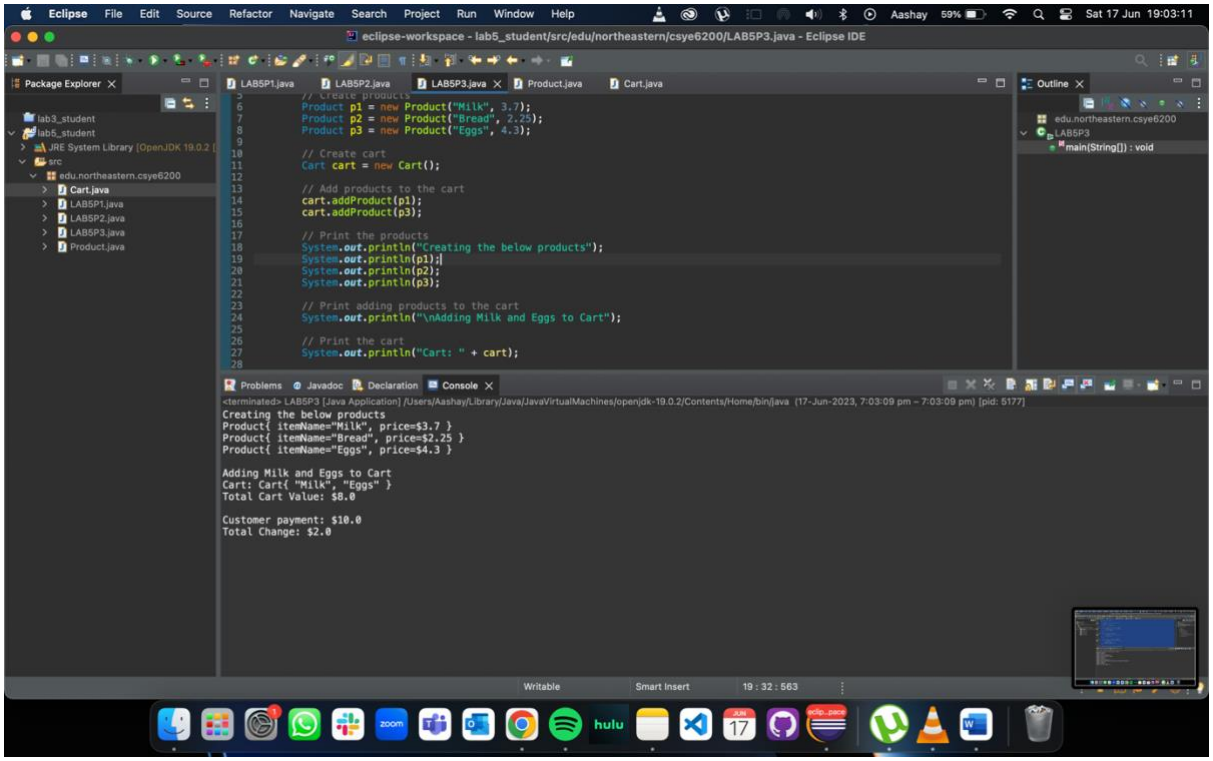
Output:

```
Creating the below products
Product{ itemName="Milk", price=$3.7 }
Product{ itemName="Bread", price=$2.25 }
Product{ itemName="Eggs", price=$4.3 }

Adding Milk and Eggs to Cart
Cart: Cart{ "Milk", "Eggs" }
Total Cart Value: $8.0

Customer payment: $10.0
Total Change: $2.0
```

Screenshots:






Eclipse File Edit Source Refactor Navigate Search Project Run Window Help
Aashay 59% Sat 17 Jun 19:03:11
eclipse-workspace - lab5_student/src/edu/northeastern/csye6200/LAB5P3.java - Eclipse IDE
Package Explorer
lab3_student
lab5_student
JRE System Library [OpenJDK 19.0.2
src
edu.northeastern.csye6200
Cart.java
LAB5P1.java
LAB5P2.java
LAB5P3.java
Product.java
LAB5P1.java LAB5P2.java LAB5P3.java Product.java Cart.java
// Create products
Product p1 = new Product("Milk", 3.7);
Product p2 = new Product("Bread", 2.25);
Product p3 = new Product("Eggs", 4.3);
// Create cart
Cart cart = new Cart();
// Add products to the cart
cart.addProduct(p1);
cart.addProduct(p3);
// Print the products
System.out.println("Creating the below products");
System.out.println(p1);
System.out.println(p2);
System.out.println(p3);
// Print adding products to the cart
System.out.println("\nAdding Milk and Eggs to Cart");
// Print the cart
System.out.println("Cart: " + cart);
Outline
edu.northeastern.csye6200
LAB5P3
main(String[]) : void
Problems Javadoc Declaration Console
<terminated> LAB5P3 [Java Application] /Users/Aashay/Library/Java/JavaVirtualMachines/openjdk-19.0.2/Contents/Home/bin/java (17-Jun-2023, 7:03:09 pm – 7:03:09 pm) [pid: 5177]
Creating the below products
Product{ itemName="Milk", price=$3.7 }
Product{ itemName="Bread", price=$2.25 }
Product{ itemName="Eggs", price=$4.3 }
Adding Milk and Eggs to Cart
Cart: Cart{ "Milk", "Eggs" }
Total Cart Value: $8.0
Customer payment: $10.0
Total Change: $2.0
Writable Smart Insert 19 : 32 : 563