Lab Assignment 03

HOMEWORK

Task 1

Design the **Toy** class in such a way that it produces the following output

Driver Code	Expected Output

`public class ToyTester{`
`  public static void main(String[] args){`
`  Toy t1 = new Toy("Car", 230);`
`  System.out.println("1================");`
`  t1.updatePrice(250);`
`  System.out.println("2================");`
`  System.out.println(t1.name);`
`  t1.showPrice();`
`  System.out.println("3================");`
`  Toy t2 = new Toy("Robot", 450);`
`  System.out.println("4================");`
`  t2.updateName("Autobot");`
`  System.out.println("5================");`
`  System.out.println(t2.name);`
`  t2.showPrice();`
`  }`
`}`
	`A new toy has been made!`
	`1================`
	`2================`
	`Car`
	`price: 250 Taka`
	`3================`
	`A new toy has been made!`
	`4================`
	`Changing old name: Robot`
	`new name: Autobot`
	`5================`
	`Autobot`
	`price: 450 Taka`

Task 2

Complete the following **Cart** class to generate the given output from the tester code:

- A cart will have a cart number which will be assigned in ***create_cart()*** method.
- Each cart can hold up to 3 items (at max).
- Each cart must have two arrays to store items and their respective prices.
- The items inside a cart will be added in ***addItem()*** method only if the cart items do not exceed 3.
- The ***giveDiscount()*** method saves the discount given to that cart object and updates the price accordingly.

Driver Code	Output

public class CartTester{
  public static void main(String [] args){
  Cart c1 = new Cart ();
  Cart c2 = new Cart ();
  Cart c3 = new Cart ();
 	
  c1.create_cart(1);
  c2.create_cart(2);
  c3.create_cart(3);
  System.out.println("====1====");
  c1.addItem("Table", 3900.5);
  c1.addItem("Chair", 1400.76);
  c1.addItem(5400.87, "Television");
  c1.addItem(5000.0, "Refrigerator");
 	
  System.out.println("====2====");
  c2.addItem("Stove",439.90);
 	
  System.out.println("====3====");
  c3.addItem("Chair",1400.5);
  c3.addItem(3400.0, "Chair");
 	
  System.out.println("====4====");
  c1.cartDetails();
 	
  System.out.println("====5====");
  c2.cartDetails();
 	
  System.out.println("====6====");
  c3.cartDetails();
  c1.giveDiscount(10);
 	
  System.out.println("====7====");
  c1.cartDetails();
  }
}
	====1====
	Table added to cart 1.
	You have 1 item(s) in your cart now.
	Chair added to cart 1.
	You have 2 item(s) in your cart now.
	Television added to cart 1.
	You have 3 item(s) in your cart now.
	You already have 3 items on your cart
	====2====
	Stove added to cart 2.
	You have 1 item(s) in your cart now.
	====3====
	Chair added to cart 3.
	You have 1 item(s) in your cart now.
	Chair added to cart 3.
	You have 2 item(s) in your cart now.
	====4====
	Your cart(c1) :
	Table - 3900.5
	Chair - 1400.76
	Television - 5400.87
	Discount Applied: 0.0%
	Total price: 10702.130000000001
	====5====
	Your cart(c2) :
	Stove - 439.9
	Discount Applied: 0.0%
	Total price: 439.9
	====6====
	Your cart(c3) :
	Chair - 1400.5
	Chair - 3400.0
	Discount Applied: 0.0%
	Total price: 4800.5
	====7====
	Your cart(c1) :
	Table - 3900.5
	Chair - 1400.76
	Television - 5400.87
	Discount Applied: 10.0%
	Total price: 9631.917000000001

Task 3

Design the **Reader** class in such a way so that the following code provides the expected output.

- A reader will have a name, capacity to read and an array of books they are reading.
- The initial capacity of a reader will be 2. The initial name will be "New user".

Driver Code	Expected Output

`public class Reader_tester {`
`  public static void main(String[] args){`
 	
`    System.out.println("1 ==========");`
`    Reader r1 = new Reader("Messi");`
`    Reader r2 = new Reader("Ronaldo", 3);`
`    System.out.println("2 ==========");`
`    r1.readerInfo();`
`    System.out.println("3 ==========");`
`    r2.addBook("Java");`
`    r2.addBook("Python");`
`    r2.addBook("C++");`
`    r2.readerInfo();`
`    System.out.println("4 ==========");`
`    r1.addBook("C#");`
`    r1.addBook("Rust");`
`    r1.addBook("GoLang");`
`    System.out.println("5 ==========");`
`    r2.addBook("Python");`
`    System.out.println("6 ==========");`
`    r1.readerInfo();`
`    System.out.println("7 ==========");`
`    r1.updateCapacity(3);`
`    System.out.println("8 ==========");`
`    r1.addBook("GoLang");`
`    System.out.println("9 ==========");`
`    r1.readerInfo();`
 	
`  }`
`}`
	`1 ==========`
	`A new reader is created!`
	`A new reader is created!`
	`2 ==========`
	`Name: Messi`
	`Capacity: 2`
	`Books:`
	`No books added yet`
	`3 ==========`
	`Name: Ronaldo`
	`Capacity: 3`
	`Books:`
	`Book 1: Java`
	`Book 2: Python`
	`Book 3: C++`
	`4 ==========`
	`No more capacity`
	`5 ==========`
	`No more capacity`
	`6 ==========`
	`Name: Messi`
	`Capacity: 2`
	`Books:`
	`Book 1: C#`
	`Book 2: Rust`
	`7 ==========`
	`Capacity has changed to 3`
	`8 ==========`
	`9 ==========`
	`Name: Messi`
	`Capacity: 3`
	`Books:`
	`Book 1: C#`
	`Book 2: Rust`
	`Book 3: GoLang`

Task 4

You are building a tracker system that will keep track of a person's income and expenses.

- When the ***createTracker()*** method is invoked it sets the balance to 1.0 taka.
- The ***info()*** method **returns** a String with the trackers information.
- If the total balance becomes 0 after the ***expense()*** method is called it prints "You're broke!" . Again if the available balance is less than the expense it prints "Not enough balance.". Otherwise the method prints "Balance updated" after updating the balance.
- The last expense and income history can be seen by using the ***history()*** method.

Driver Code	Output

`public class Tester4{`
`  public static void main(String[] args) {`
`  MoneyTracker tr1 = new MoneyTracker();`
`  System.out.println(tr1.info());`
`  tr1.createTracker("John");`
`  System.out.println("1 ==========");`
`  System.out.println(tr1.info());`
`  System.out.println("2 ==========");`
`  tr1.income(1000);`
`  System.out.println(tr1.info());`
`  System.out.println("3 ==========");`
`  tr1.expense(800);`
`  tr1.expense(100);`
`  System.out.println(tr1.info());`
`  System.out.println("4 ==========");`
`  tr1.showHistory();`
`  System.out.println("5 ==========");`
`  tr1.expense(101);`
`  System.out.println("6 ==========");`
`  tr1.expense(200);`
`  System.out.println("7 ==========");`
`  tr1.income(200);`
`  tr1.showHistory();`
`  System.out.println("8 ==========");`
`  }`
`}`
	`A new money tracker has been launched.`
	`Name: null`
	`Current Balance: 0.0`
	`1 ==========`
	`Name: John`
	`Current Balance: 1.0`
	`2 ==========`
	`Balance Updated!`
	`Name: John`
	`Current Balance: 1001.0`
	`3 ==========`
	`Balance Updated.`
	`Balance Updated.`
	`Name: John`
	`Current Balance: 101.0`
	`4 ==========`
	`Last added: 1000.0`
	`Last spent: 100.0`
	`5 ==========`
	`You're broke!`
	`6 ==========`
	`Not enough balance.`
	`7 ==========`
	`Balance Updated!`
	`Last added: 200.0`
	`Last spent: 101.0`
	`8 ==========`

Task 5

```
1  public class A{
2      public int x, y, z = 5;
3      public double p = 0.0;
4      public void methodA(int x, int m) {
5          this.x = methodB(this.x);
6          p = x + this.x % m * 3.0;
7          y = y + methodB(x++, this.x);
8          System.out.println(this.x +" " + x + y + " " + p) ;
9      }
10     public int methodB(int q, int n) {
11         int arr[] = {3,4,5};
12         arr[0] = arr[0] + this.x + n;
13         arr[1] = q + arr[1];
14         System.out.println(arr[0] +" " + arr[1] + " " + arr[2]) ;
15         return arr[1] + arr[2];
16     }
17     public int methodB(int y) {
18         if(y % 2 == 0) {
19             int temp = this.methodB(2, y);
20             return temp;
21         }
22         else{
23             return 4;
24        }
25     }
26 }
```

Driver Code	Output		

public class Tester11 {
  public static void main(String [] args){
  A a1;
  a1 = new A();
  a1.methodA(2,3);
  a1.methodB(5,4);
  }
}			

Task 6

```
1  public class Maze{
2          public int x;
3          public Maze(){
4            int x = 17;
5            this.methodB(x, -this.x);
6          }
7          public void methodA(){
8             int m = 0, x = 9;
9              m = methodB(m-3)+x;
10             this.x = ++x;
11            System.out.println(this.x+" "+m);
12            this.methodB(x,m);
13             System.out.println(x+" "+(m+this.x));
14             methodB(m);
15         }
16         public int methodB(int y){
17             x=y*y;
18             System.out.println(x+" "+y);
19            return x-11;
20         }
21         public void methodB(int z, int x){
22             z=z-2;
23             x=this.x-2*x;
24             System.out.println(z+" "+this.x);
25         }
26 }
```

DRIVER CODE	OUTPUTS	
public class MazeTester{		
public static void main(String args []){		
Maze m1 = new Maze();		
m1.methodA();		
new Maze().methodB(7,15);		
}		
}		