Workshop #4: Inheritance

Step 1: Create a package named “**DTO**”, it contains some files: Item.java, Vase.java, Statue.java, and Painting.java

Step 2: Create another package named “**GUI**”, it contains the AntiqueShop.java file

Implement the class diagram as follows:

|  |
| --- |
| Item |
| #value: int #creator: String |
| +Item() +Item(int, String) +getters/setters +output():void +input():void |

|  |
| --- |
| Vase |
| -height: int -material: String |
| +Vase() +Vase(int, String, int, String) +getters/setters +outputVase():void +InputVase(): void |

|  |
| --- |
| Statue |
| -weight: int -colour: String |
| +Statue() +Statue(int, String, int, String) +getters/setters +outputStatue():void +inputStatue():void |

|  |
| --- |
| Painting |
| -height: int -width: int -isWatercolour: boolean -isFramed: boolean |
| +Painting() +Painting(int, String, int, int, boolean, boolean ) +getters/setters +outputPaiting():void +inputPainting():void |

|  |
| --- |
| AntiqueShop |
|  |
| +main():void |

This is an “**association**” relationship and simply denotes that AntiqueShop is making use of Vase, Statue, and Painting, in the sense that it has declared references to them, and thus there is a dependency.

**Requirement**:  
 1. In the file Item.java,

* The method input(): Using Scanner class to input all fields of the Item class. Verify: value>0, creator is not empty
* The method output(): print out all fields

2. In the file Vase.java,

* The method inputVase(): Using Scanner class to input all fields of the Vase class.
* The method outputVase(): print out all fields of the Vase class

*Hint:*

public class Vase{  
 …  
 public void inputVase(){  
 input(); // call the inherited method to input two fields: value, creator

Scanner sc=new Scanner(System.in);  
 System.out.println(“Input a height:”);  
 height=sc.nextInt();  
 System.out.println(“Input a material:”);

sc=new Scanner(System.in);  
 material =sc.nextLine();  
  
 }  
 public void outputVase(){  
 output(); // call the inherited method to print two fields out: value, creator

System.out.println(“Height:” + height);  
 System.out.println(“Material:”+ material);  
 }

…  
}

1. You do the same for Statue class, Painting class
2. In the file “Test.java”. you type like as follow:

public class Tester{

public static void main(String[] args){  
 Item item=null;

int choice=0;

Scanner sc=….

do{  
   
 System.out.println(“1. Create a Vase:”);

System.out.println(“2. Create a Statue:”);

System.out.println(“3. Create a Painting:”);

System.out.println(“4. Display the Item:”);

System.out.println(“Input a choice:”);

Choice=sc.nextInt();  
 switch(choice){  
 case 1:  
 item=new Vase();  
 ((Vase)item).inputVase();

break;   
 case 2:  
 item =new Statue();  
 ((Statue) item).inputStatue();

break;

case 3:

item =new Painting();  
 ((Painting) item).inputPainting();

break;

case 4:  
 if(item!=null){  
 if(item instanceof Vase)  
 ((Vase) item).outputVase();  
 else if(item instanceof Statue)  
 ((Statue) item).outputStatue ();  
 else if(item instanceof Painting)  
 ((Painting) item).outputPainting ();   
 }  
 else System.out.println(“ you need to create an object”);

break;  
 }  
  
 }while(choice<=4);  
  
 }

}