Workshop #4: Inheritance

**Learning Outcomes:**

Upon successful completion of this workshop, you will have demonstrated the abilities to:

* Design and implement classes in the “is-a” relationship.
* Practice casting
* Describe to your instructor what you have learned in completing this workshop.

**Requirements:**

**Part 1:** [7 points]

To complete this task you should read and study the lecture [Inheritance](../index.html)

Step 1: Create a new project named “**ItemManager**”.

Step 2: Create a package named “**DTO**”, it contains some files: Item.java, Vase.java, Statue.java, and Painting.java

Step 3: 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 |

The AntiqueShop class 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{  
 …

//this method is used to input all fileds of a vase object  
 public void inputVase(){  
 input(); // call the inherited method to input two fields: value, creator

//TODO: you is required to add more your code to input two fields : height, material

// use try..catch/throws to handle exceptions  
 }

//this method displays information of a vase object  
 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 “AntiqueShop.java”. you type like as follow:

public class AntiqueShop {

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); }

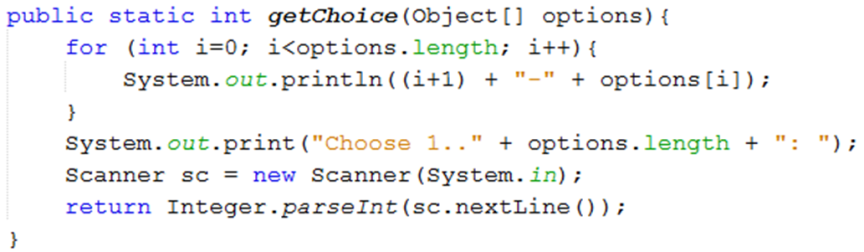
}

1. (Optional) Now, you is required to update the above program. You should create a new class named **Menu**. This class contains one static method

*//use this method to show pre-defined options*

*//input: an array contains the list of options*

*//output: return a user’s choice that is inputted from the keyboard.*



* Update the main method to use the Menu class.

public class AntiqueShop {

public static void main(String[] args){

String[] options={“ Create a Vase “,”Create a Statue”,” Create a Statue”,” display the item”};  
 Item item=null;

int choice=0;

do{  
   
 choice=Menu.getChoice(options);

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

break;   
 ….  
 }while(…);

}

**Part 2: Draw the memory map when the program runs [3 points]**

Explain step by step what happened when the program runs and answer some questions.

Step 1: print the menu and get choice from user

Step 2: With each selection, program will run each method

Step 3: Run step 1

* What is stored in the static heap, stack, dynamic heap?

*+ Static heap: the class and static variables are stored*

*+ Stack: method calls, local variables and reference variables*

*+ Dynamic heap: the objects are stored*

* What are objects in the program?

*Item and sc*

* What is the item variable storing?

*It store the address of object Vase, Statue , Painting*

* Why must you cast to call the method inputVase()/outputVase()?

*Because the reference variable “item” is defined on Item class*

* What is the error thrown when you cast it wrong?

*ClassCastException*

* What methods can you call if you don’t cast the item variable?

*Input and output*