Advanced Programming (CSE201), Quiz -1

Time allocated: 04:15pm – 4:35pm (20 minutes)

Instructions:

- This is a closed book quiz.
- Only reasonable and clearly mentioned assumptions (if any) would be accepted.
- For justifications, please be as concise as possible (2-3 sentences only)
- IIIT plagiarism policy is applicable if any such cases found
- Write your answers on a plain sheet that you can upload by taking a picture of the same (ensure low resolution so that the upload size is smaller)
- You can email the quiz solutions to **"ap-m2020-submission@iiitd.ac.in"**. Subject of the mail should be **<Rollno>_Quiz1**
- We will not consider any submission that is emailed beyond 4.45pm. It is your responsibility to ensure you email it on time. Please ensure you have proper internet connectivity as we are giving you sufficient extra time to send the email.

NOTE: Answers in red font and rubric in blue font.

Question-1) Read the below mentioned problem description and answer below questions:

"A petshop has a permanent groomer. Whenever a dog is received by the petshop, it first lets the dog bark and jump. After this it hands over the dog to the groomer for grooming. The groomer is a busy person and so he should be able to check the petstore timings as it always keeps changing."

Implement the Object Oriented Implementation of the above program description and identify the class relationships (if any). No need to code the main method. **Pseudocode implementation is sufficient as long as it can capture all OOP requirements**. **[5 marks]**

Answers:

```
class PetShop {
private Groomer my_groomer;
public PetShop() {
my_groomer= new Groomer(this);
}
public void process(Dog d) {
d.bark();
d.jump();
my_groomer.groom(d);
}
11. public void getTiming() {
    ...
12.  }
13. }
14. class Groomer {
15. private PetShop myShop;
16. public Groomer(PetShop p) {
17. myShop = p;
18. }
19. public void groom(Dog d) {
    ....
20. }
21. }
```

Line-1:	**[0.25 marks]**
Line-2:	**[0.25 marks]**
Line-4:	**[0.25 marks]**
Line-6:	**[0.25 marks]**
Line-7&8:	**[0.25 marks]**
Line-9:	**[0.25 marks]**
Line-11:	**[0.25 marks]**
Line-14:	**[0.25 marks]**
Line-15:	**[0.25 marks]**
Line-16:	**[0.25 marks]**
Line-17:	**[0.25 marks]**
Line-19:	**[0.25 marks]**

Relationships:

A petshop has a permanent groomer => PetShop class has Groomer type field that is instantiated inside PetShop constructor => PetShop "contains" Groomer => Composition relationship (+0.66 marks)

Dog is received by the petshop => PetShop doesn't knows-about Dog, but just uses it for service => Dependency relationship (+0.67 marks)

He should be able to check the petstore timings => Groomer "knows-about" PetShop => Groomer has a field of PetShop type that is assigned using a parameterized constructor => Association relationship (+0.67 marks)

Question-2: Consider the small variation in the description given in Question-1:

"The petshop provides grooming services for all pets with a fur coat such as cat, dog and rabbits. Although each of these pets has a different type of fur coat, the permanent groomer can groom them all."

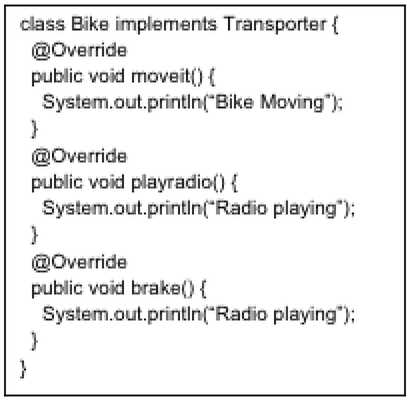
Implement the Object Oriented Implementation of the above description by modifying your original code in Question-1. You can show the code for just one of the pets other than the changes you require in your solution for Question-1. **Pseudocode implementation is sufficient as long as it can capture all OOP requirements. [2 marks]**

Answers:

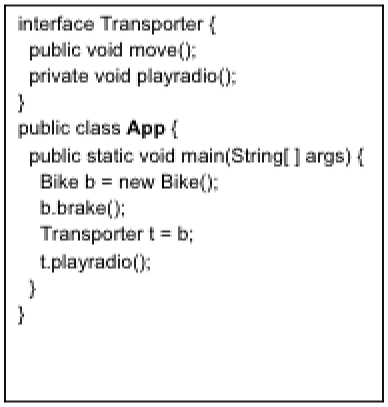
```
1.  interface Fur {
2.  public void getFurType();
3.  }
4.  class Cat implements Fur {
5.  @Override
6.  public void getFurType() {
7.  System.out.println("meau
    meau");
8.  }
9.  }
10. public class Groomer {
11. public void groom(Fur pet) {
12. pet.getFurType();
13. }
14. }
15. public class PetShop {
    ....
16. public void process(Fur pet) {
17. my_groomer.groom(pet);
18. }
19. }
```

Line-1: [0.25 marks]
Line-2: [0.25 marks]
Line-4: [0.25 marks]
Line-6: [0.25 marks] //must provide SOME DUMMY OR EMPTY implementation
Line-11: [0.25 marks]
Line-12: [0.25 marks]
Line-16: [0.25 marks]
Line-17: [0.25 marks]

Question-3) Correct the program such that it works properly **[3 marks]**



```
class Bike implements Transporter {
  @Override
  public void moveit() {
    System.out.println("Bike Moving");
  }
  @Override
  public void playradio() {
    System.out.println("Radio playing");
  }
  @Override
  public void brake() {
    System.out.println("Radio playing");
  }
}
```



```
interface Transporter {
  public void move();
  private void playradio();
}
public class App {
  public static void main(String[ ] args) {
    Bike b = new Bike();
    b.brake();
    Transporter t = b;
    t.playradio();
  }
}
```

Answer:

Bike class:
1) Compiler error for Overriding moveit method (+0.66 marks)
2) Compiler error for missing move method (+0.67 marks)
3) Compiler error for Overriding brake method (+0.67 marks)

Transporter interface:
1) Compiler error on private method   (+1 marks)