SE3001 – Software Const. & Dev.
Time Allowed: 15 mins
Section: ______________________
Roll no: ______________________

Fall 2024
Date: Sep 10, 2024
Marks Obtained:
Total Marks: 12

Quiz# 1

Q.1) You have the following code. Remove compile time errors from it. (1 mark)

```
abstract class Animal {
    protected double strength;
    abstract public void speak();
    public Animal() {
        strength = 0;
    }
    public double getStrength() {
        return this.strength;
    }
    public Animal(double strength) {
        this.strength = strength    ;
    }

    @Override
    public String toString() {
        return strength + ""; // a trick to convert into string
    }
}
```

Q.2) The following class wants to inherit from the `Animal` class.

```
class Dog {
    protected int breed; // not a good idea but anyways
    public Dog(double strength, int breed) {
      ……
    }
}
```

a. Modify the class. Remove all compile time errors and write the missing code in the constructor. (3 marks)

```
class Dog extends Animal {
    protected int breed; // not a good idea but anyways
    public Dog(double strength, int breed) {
      super(strength)
      this.breed = breed;
    }
    @override
    public void speak() {
      System.out.println(“woof!”);
    }
}
```

b. class `Animal` is overriding which class’s `toString()` method? (2 marks) class Object's

Q.3) You have the following interface:

```
interface Fightable {
    public boolean fights(Fightable other);
}
```

a) class Dog wants to be Fightable. Mention the code changes that you'll do in Dog. The function returns true if the caller object's strength is greater than the parameter object. Otherwise, it returns false. (3 marks)

```
class Dog extends Animal implements Fightable {
   protected int breed; // not a good idea but anyways
   public Dog(double strength, int breed) {
      super(strength);
      this.breed = breed;
   }

   @Override
   public void speak() {
      System.out.println("Woof!");
   }
   @Override
   public boolean fights(Fightable other) {
      Animal an = (Animal) other;
      return this.strength > an.getStrength();
   }
}
```

b) How can we allow the fight of a Dog with a Cat? You may mention code additions in English but do write the calling/driver code. (3 marks)

Make a subclass Cat that implements Fightable interface.

```
public static void main(String[] args) {

Fightable aDog = new Dog(4, 8);
      Fightable aCat = new Cat(5, 8);
      boolean result = aDog.fights(aCat);
      ~~System.out.println(result);~~
   }
```