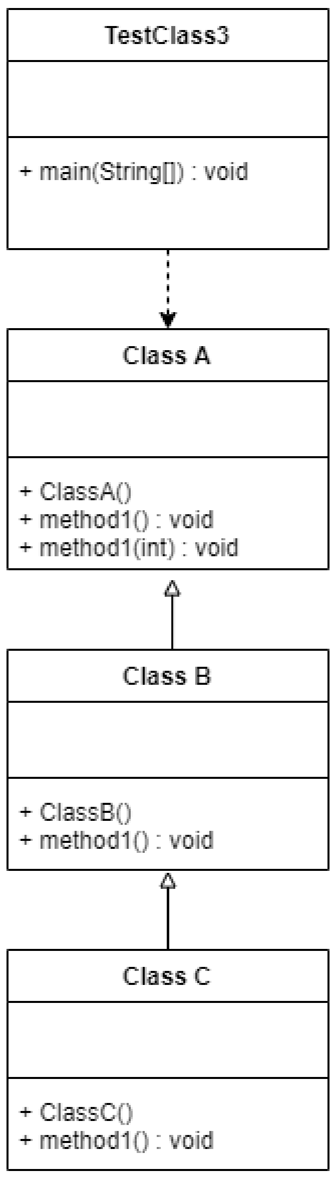
Page 238 Exercise 1 – Concept

Q1

i.



ii.

a. Class C
b. Class B
c. Class B
d. Class C

Q3

i.

a. False
b. True
c. True
d. False
e. True

f. True
g. False

ii.
a. No
b. Yes

iii.
a. Yes
b. No

Q8

The printMessage method in the parent class (in program 9.15) is declared as final, which its' child class, ProfNetBallPlayer which also has a printMessage method fails to override the parent's method.

Amended Program:

```
//Program 9.15

public class NetBallPlayer{

        private int jerseyNumber;

        public void printMessage(){ /////notice the method here omit the final keyword

                System.out.println("I am a good netball player");

        }

}
```

Q9

A final class cannot be extended (cannot have child class). If we're intended to extend the Game1 class, it should not be declared with final keyword.

Amended Program:

```
//Program 9.17

public class Game1{  //// notice the class declaration here omit the final keyword

        private int count;

        public void printGameRule(){

                System.out.println("Count your turn "+ count);

        }

}
```

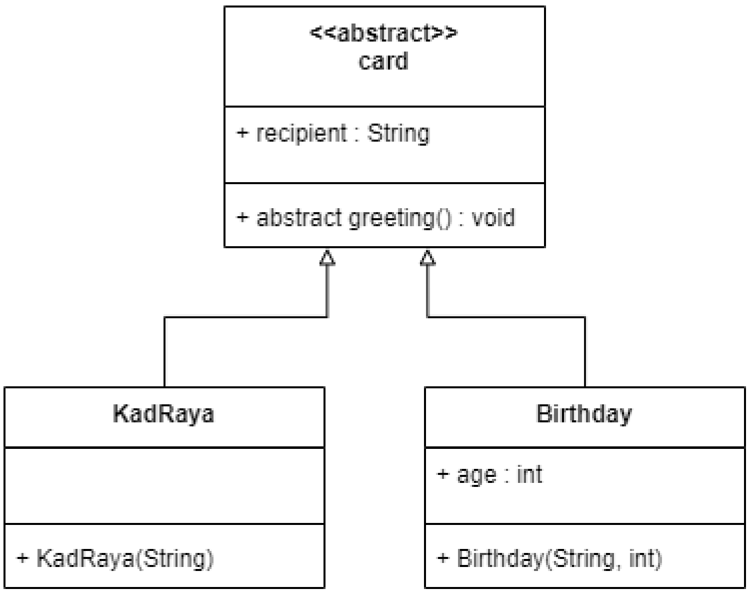
Page 250 Exercise 2 – Abstract Class

Q1.

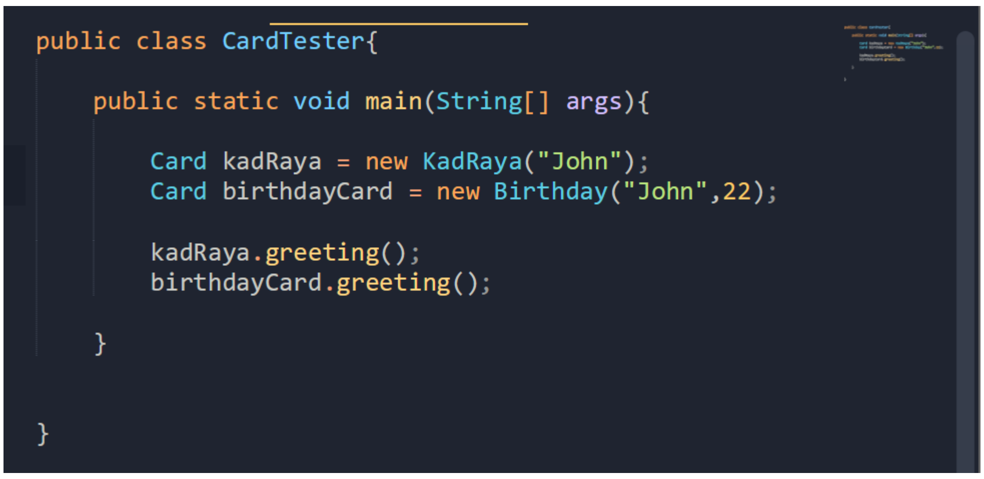
i.	Illegal
ii.	Illegal
iii.	Legal
iv.	Legal
v.	Illegal

Q2.

i.

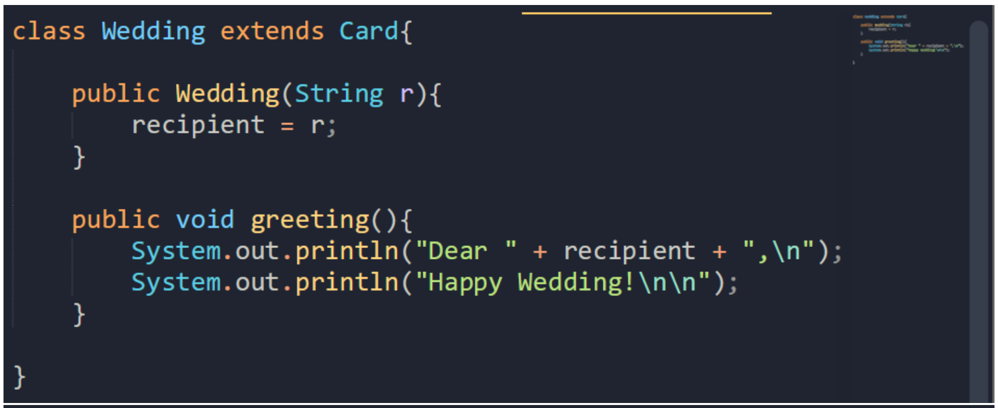


ii.

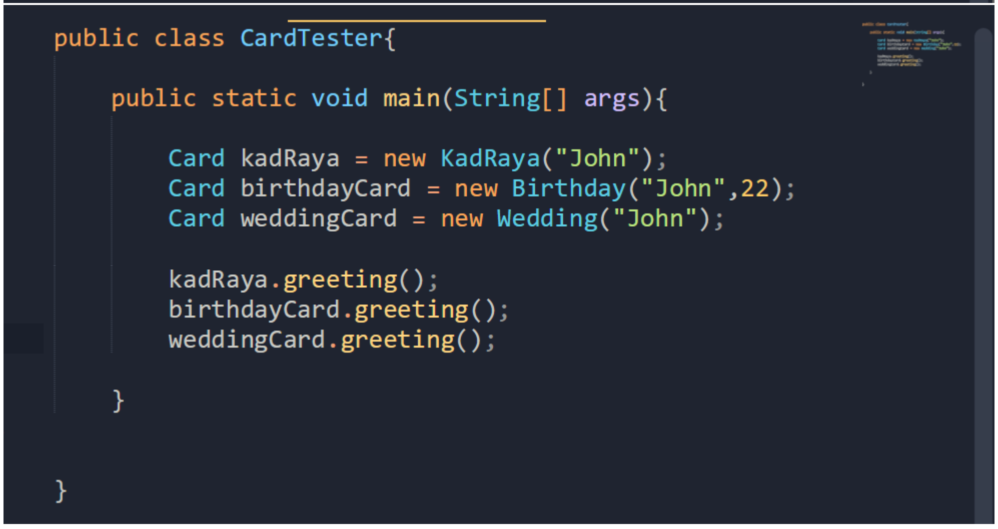


```
public class CardTester{

    public static void main(String[] args){

        Card kadRaya = new KadRaya("John");
        Card birthdayCard = new Birthday("John",22);

        kadRaya.greeting();
        birthdayCard.greeting();

    }


}
```

iii.

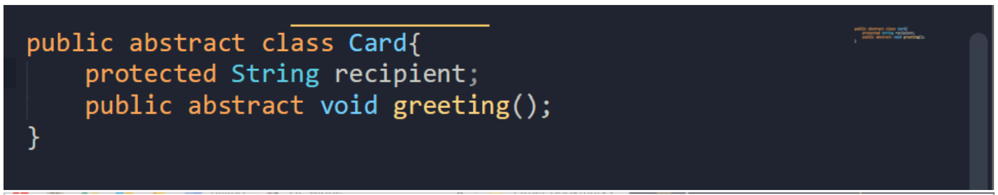


```java
class Wedding extends Card{

    public Wedding(String r){
        recipient = r;
    }

    public void greeting(){
        System.out.println("Dear " + recipient + ",\n");
        System.out.println("Happy Wedding!\n\n");
    }

}
```

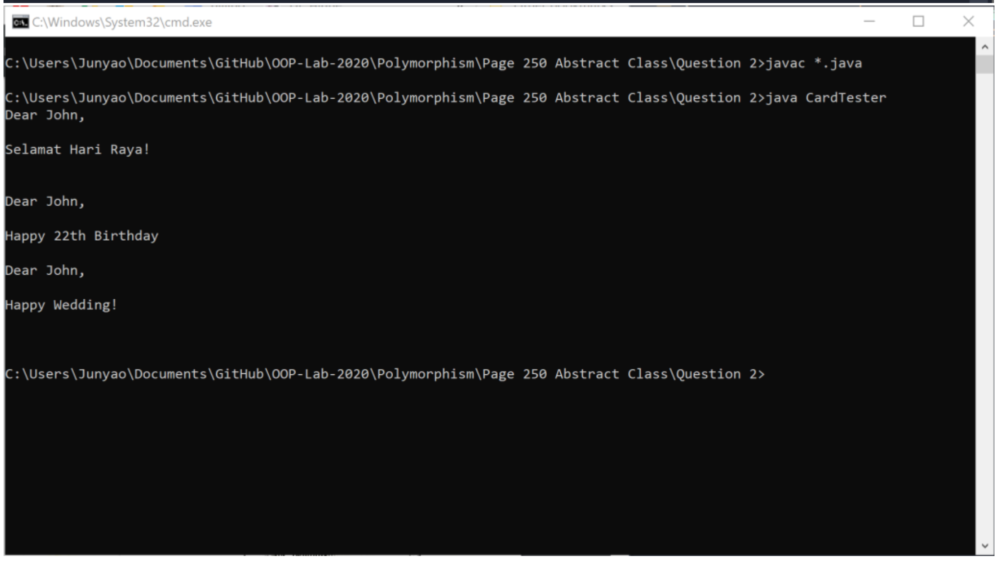


```java
   public class CardTester{

       public static void main(String[] args){

           Card kadRaya = new KadRaya("John");
           Card birthdayCard = new Birthday("John",22);
           Card weddingCard = new Wedding("John");

           kadRaya.greeting();
           birthdayCard.greeting();
           weddingCard.greeting();

       }


   }
```

iv.



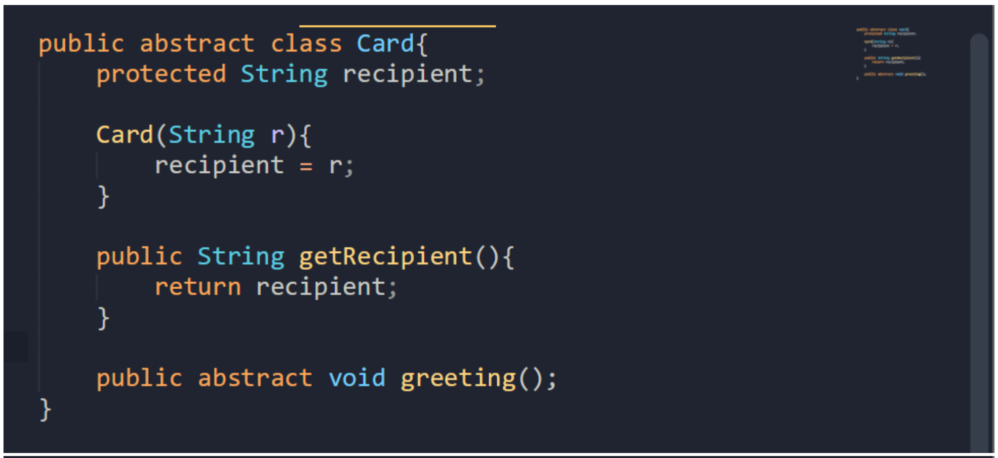
```java
public abstract class Card{
    protected String recipient;
    public abstract void greeting();
}
```



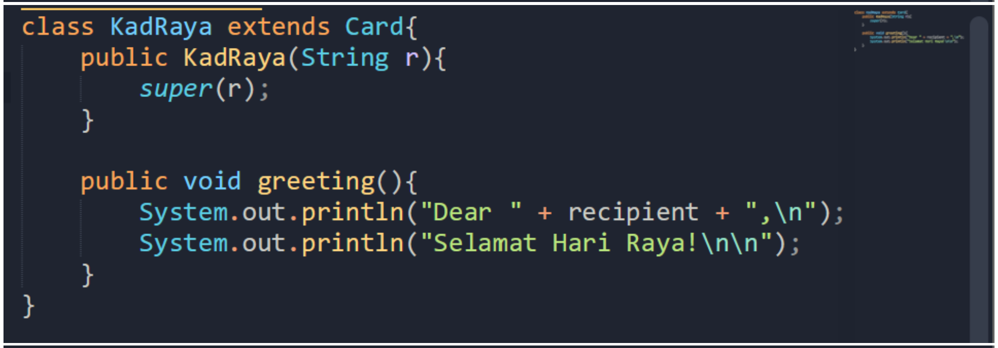
C:\Windows\System32\cmd.exe

```
C:\Users\Junyao\Documents\GitHub\OOP-Lab-2020\Polymorphism\Page 250 Abstract Class\Question 2>javac *.java

C:\Users\Junyao\Documents\GitHub\OOP-Lab-2020\Polymorphism\Page 250 Abstract Class\Question 2>java CardTester
Dear John,

Selamat Hari Raya!


Dear John,

Happy 22th Birthday

Dear John,

Happy Wedding!



C:\Users\Junyao\Documents\GitHub\OOP-Lab-2020\Polymorphism\Page 250 Abstract Class\Question 2>
```

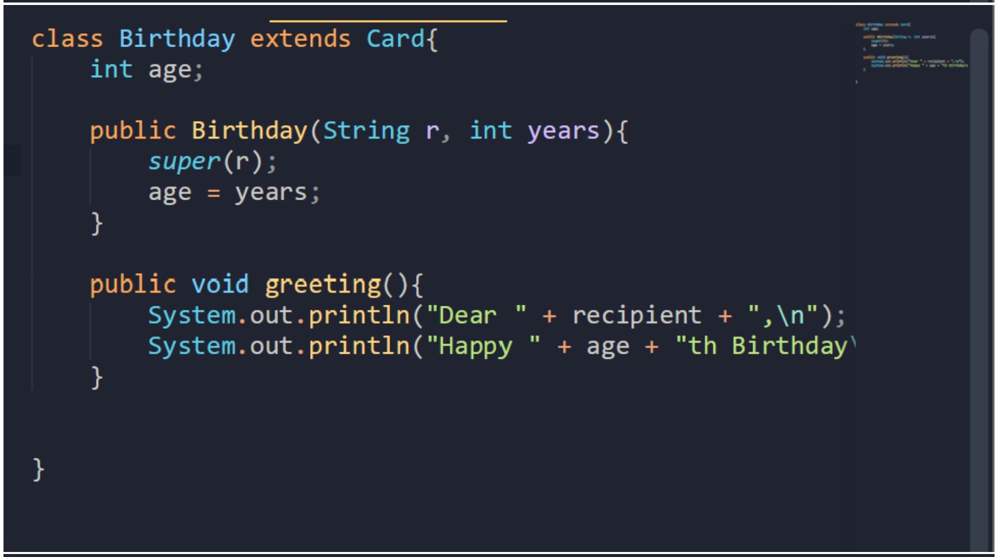
v.



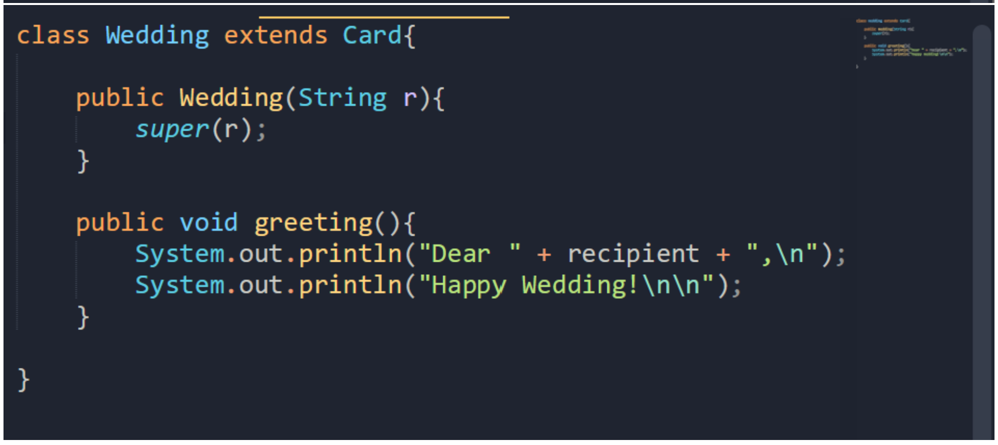
```java
public abstract class Card{
    protected String recipient;

    Card(String r){
        recipient = r;
    }

    public String getRecipient(){
        return recipient;
    }

    public abstract void greeting();
}
```



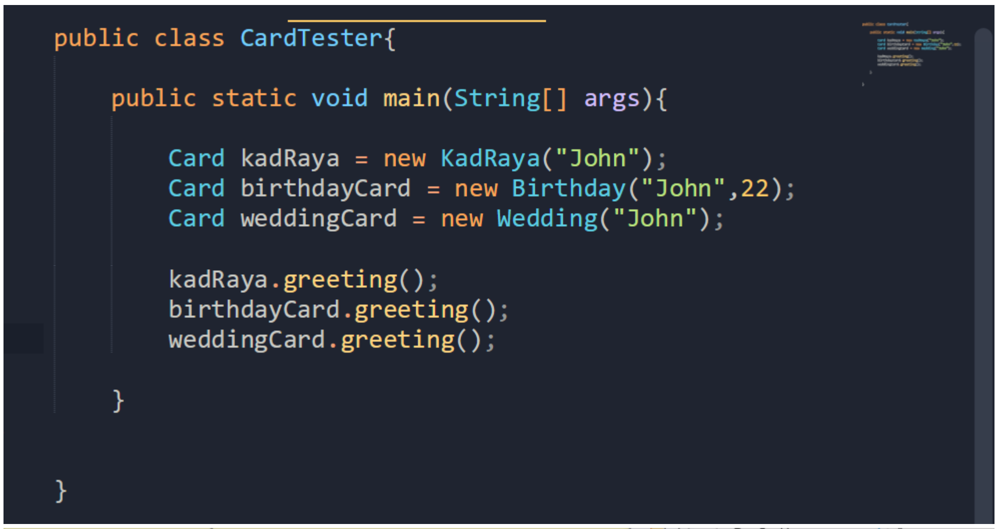
```java
class KadRaya extends Card{
    public KadRaya(String r){
        super(r);
    }

    public void greeting(){
        System.out.println("Dear " + recipient + ",\n");
        System.out.println("Selamat Hari Raya!\n\n");
    }
}
```



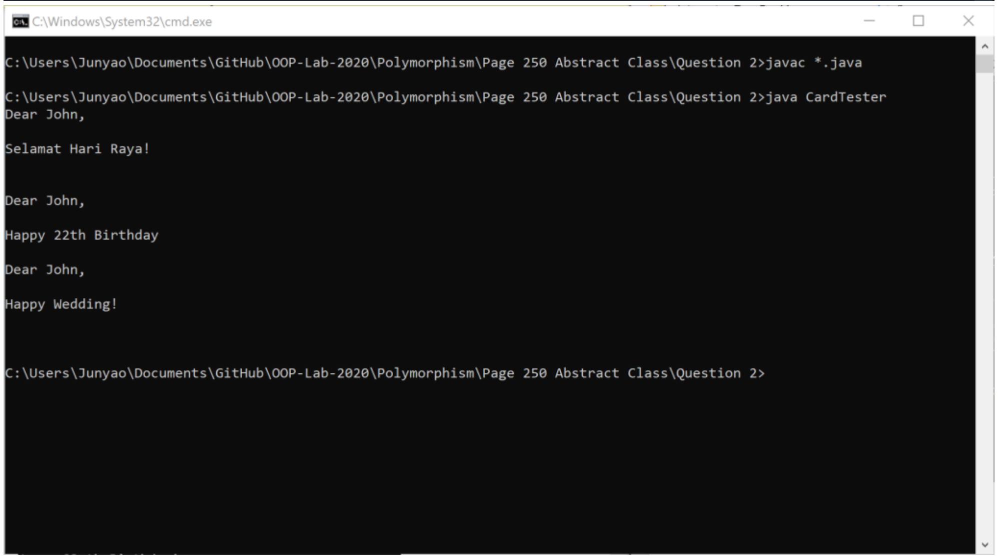
```java
class Birthday extends Card{
    int age;

    public Birthday(String r, int years){
        super(r);
        age = years;
    }

    public void greeting(){
        System.out.println("Dear " + recipient + ",\n");
        System.out.println("Happy " + age + "th Birthday
    }


}
```



```java
class Wedding extends Card{

    public Wedding(String r){
        super(r);
    }

    public void greeting(){
        System.out.println("Dear " + recipient + ",\n");
        System.out.println("Happy Wedding!\n\n");
    }

}
```



```java
public class CardTester{

    public static void main(String[] args){

        Card kadRaya = new KadRaya("John");
        Card birthdayCard = new Birthday("John",22);
        Card weddingCard = new Wedding("John");

        kadRaya.greeting();
        birthdayCard.greeting();
        weddingCard.greeting();

    }


}
```



```
C:\Users\Junyao\Documents\GitHub\OOP-Lab-2020\Polymorphism\Page 250 Abstract Class\Question 2>javac *.java

C:\Users\Junyao\Documents\GitHub\OOP-Lab-2020\Polymorphism\Page 250 Abstract Class\Question 2>java CardTester
Dear John,

Selamat Hari Raya!


Dear John,

Happy 22th Birthday

Dear John,

Happy Wedding!


C:\Users\Junyao\Documents\GitHub\OOP-Lab-2020\Polymorphism\Page 250 Abstract Class\Question 2>
```

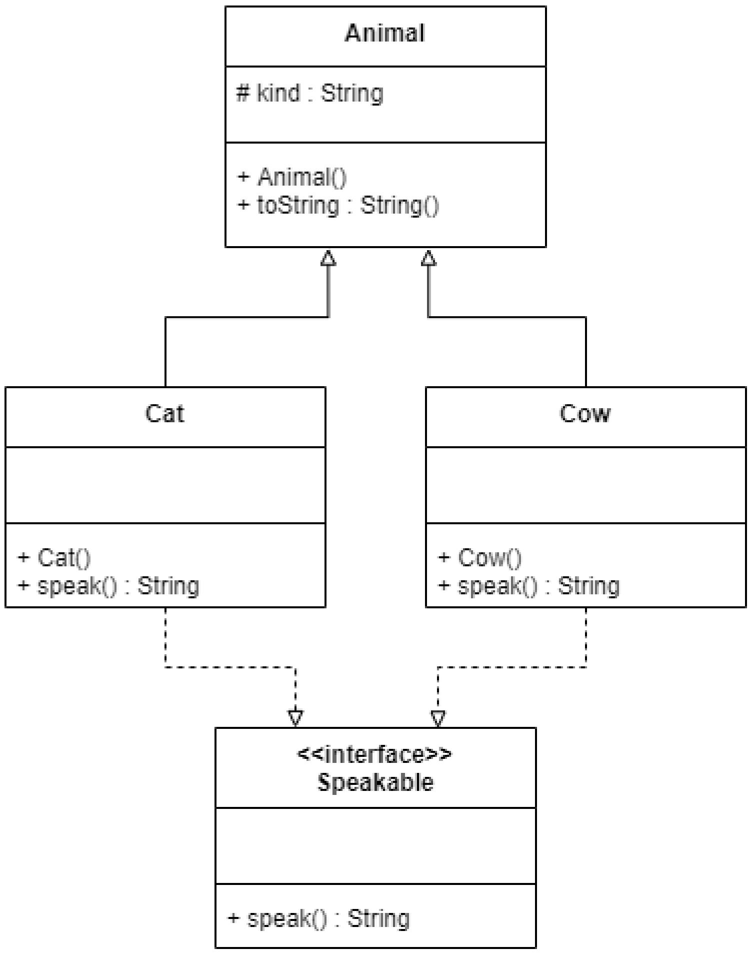
Page 259. Exercise 3 – Interface

Q1.

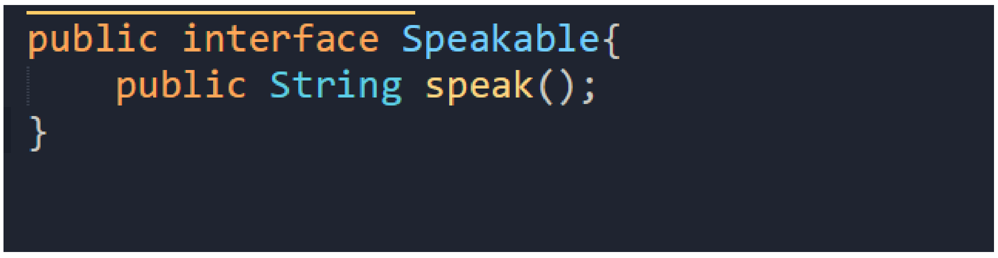
i. Legal
ii. Legal
iii. Legal
iv. Illegal

Q2

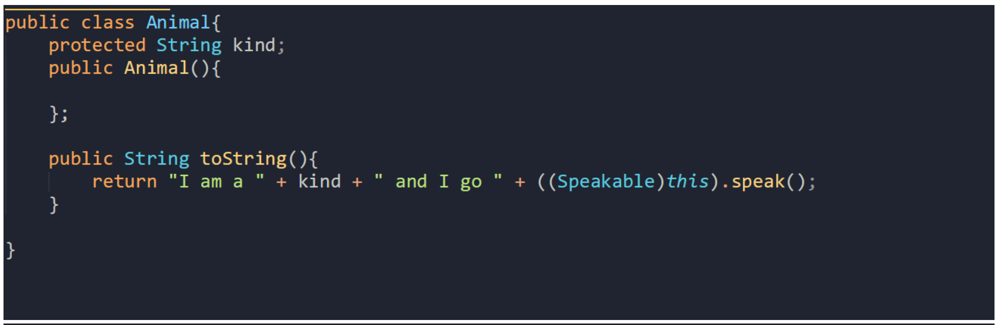
i.



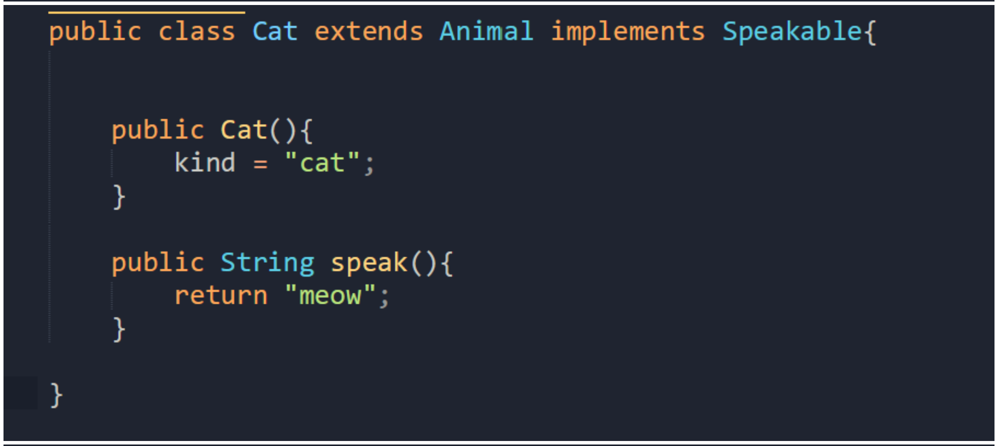
ii.



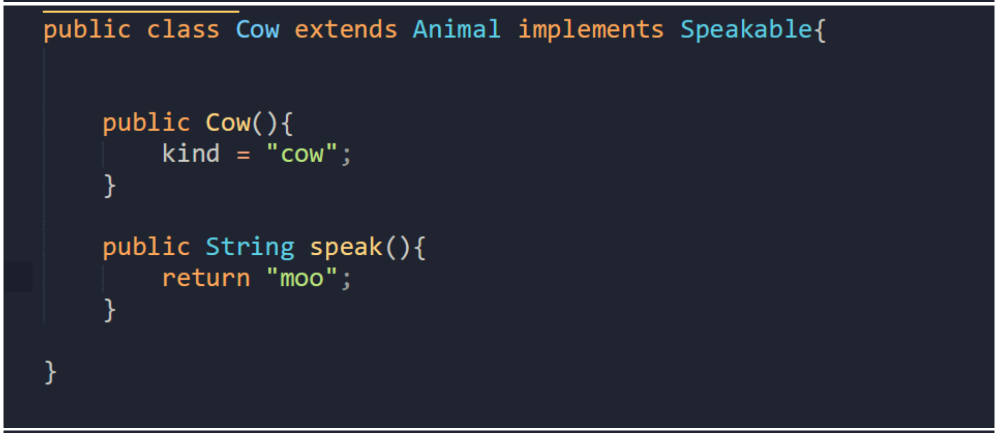
```
public interface Speakable{
    public String speak();
}
```



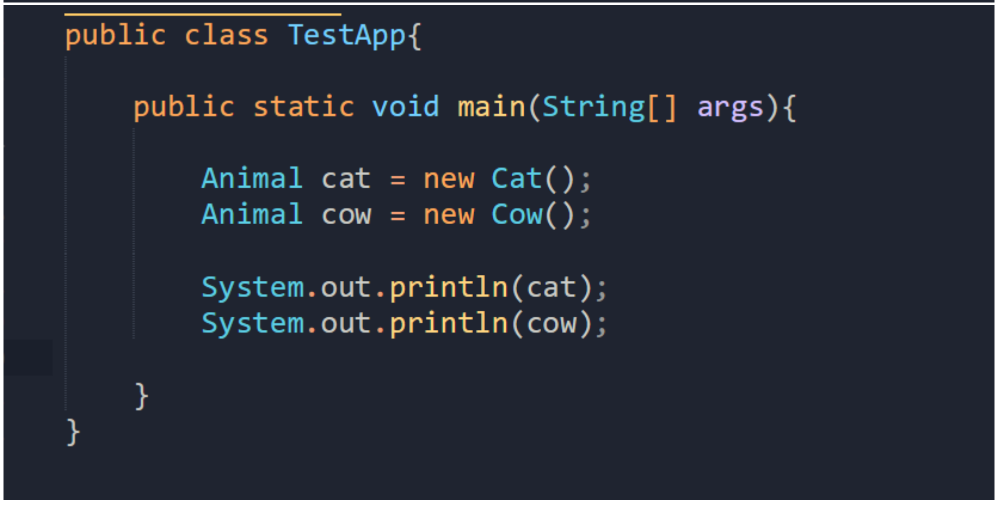
```java
public class Animal{
    protected String kind;
    public Animal(){

    };

    public String toString(){
        return "I am a " + kind + " and I go " + ((Speakable)this).speak();
    }

}
```



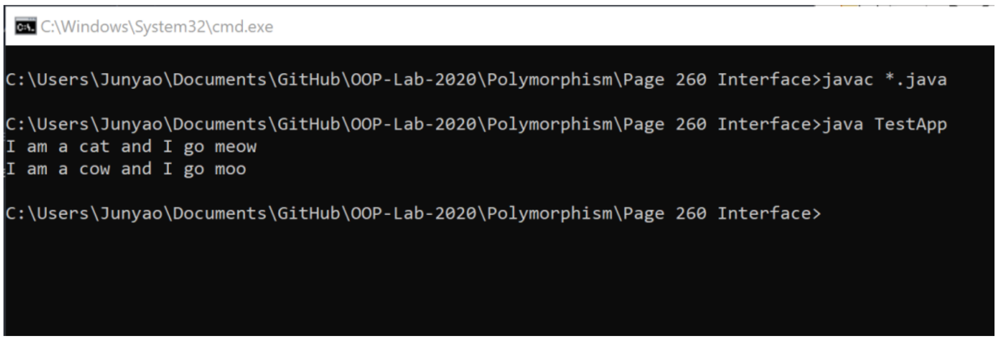
```java
public class Cat extends Animal implements Speakable{


    public Cat(){
        kind = "cat";
    }

    public String speak(){
        return "meow";
    }

}
```



```java
public class Cow extends Animal implements Speakable{


    public Cow(){
        kind = "cow";
    }

    public String speak(){
        return "moo";
    }

}
```



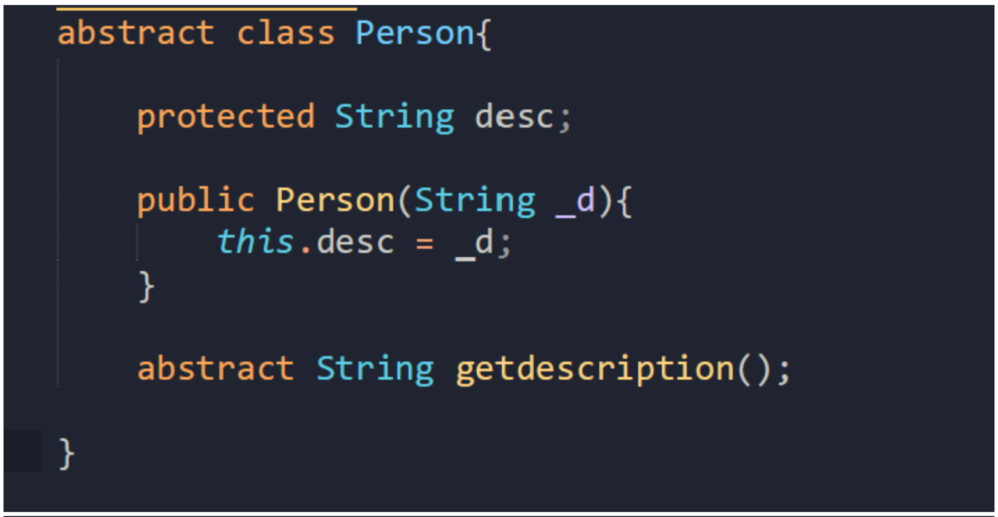
```java
public class TestApp{

    public static void main(String[] args){

        Animal cat = new Cat();
        Animal cow = new Cow();

        System.out.println(cat);
        System.out.println(cow);


    }
}
```



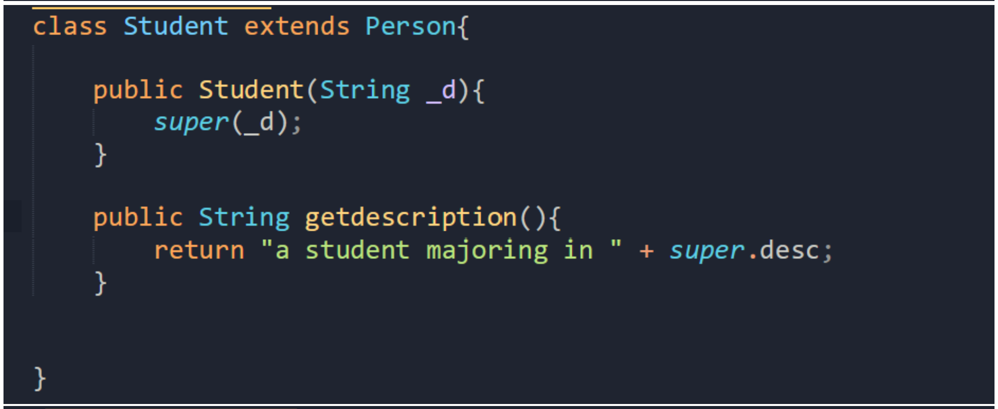
```
C:\Users\Junyao\Documents\GitHub\OOP-Lab-2020\Polymorphism\Page 260 Interface>javac *.java

C:\Users\Junyao\Documents\GitHub\OOP-Lab-2020\Polymorphism\Page 260 Interface>java TestApp
I am a cat and I go meow
I am a cow and I go moo

C:\Users\Junyao\Documents\GitHub\OOP-Lab-2020\Polymorphism\Page 260 Interface>
```

The Cow and Cat class is a child class of the Animal Class. When passed into the System.out.println, the Cow and Cat class's overridden toString method is called and return a string to be printed on the console.

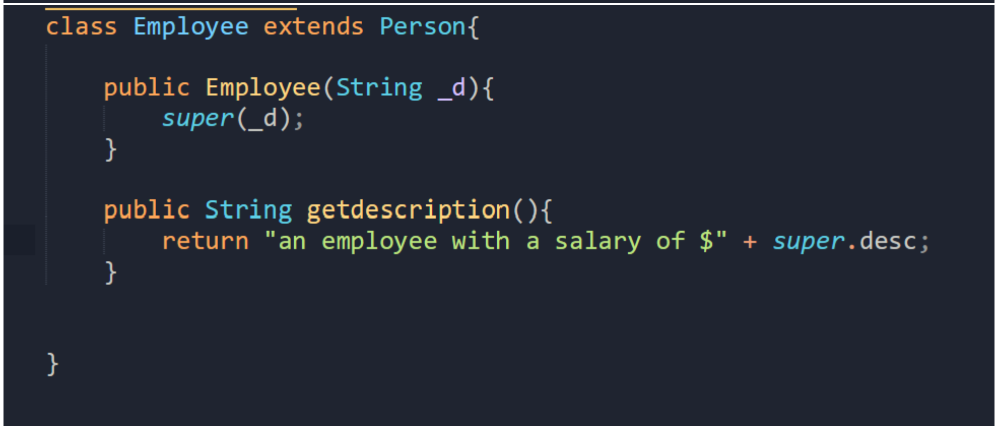
Page 263 – Problem Solving



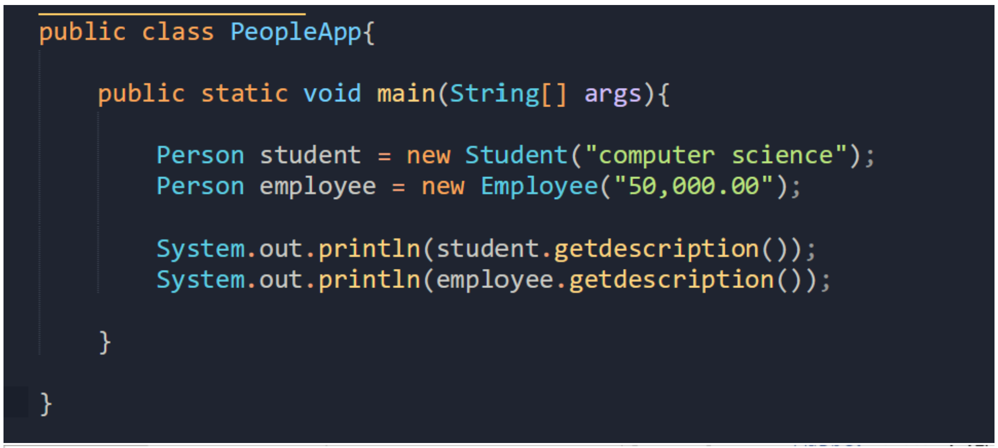
```java
abstract class Person{

    protected String desc;

    public Person(String _d){
        this.desc = _d;
    }

    abstract String getdescription();

}
```



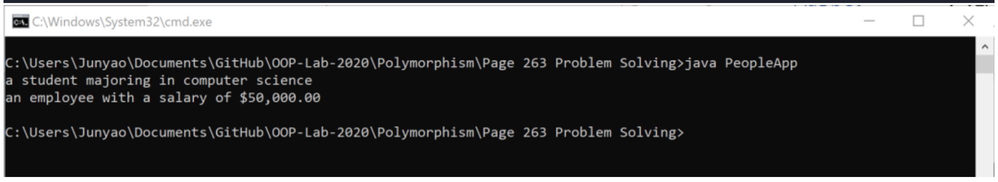
```java
class Student extends Person{

    public Student(String _d){
        super(_d);
    }

    public String getdescription(){
        return "a student majoring in " + super.desc;
    }


}
```



```java
class Employee extends Person{

    public Employee(String _d){
        super(_d);
    }

    public String getdescription(){
        return "an employee with a salary of $" + super.desc;
    }


}
```



```java
public class PeopleApp{

    public static void main(String[] args){

        Person student = new Student("computer science");
        Person employee = new Employee("50,000.00");

        System.out.println(student.getdescription());
        System.out.println(employee.getdescription());

    }

}
```



C:\Windows\System32\cmd.exe

```
C:\Users\Junyao\Documents\GitHub\OOP-Lab-2020\Polymorphism\Page 263 Problem Solving>java PeopleApp
a student majoring in computer science
an employee with a salary of $50,000.00

C:\Users\Junyao\Documents\GitHub\OOP-Lab-2020\Polymorphism\Page 263 Problem Solving>
```