CS2304 JAVA PROGRAMMING

Primitive Data types in Java

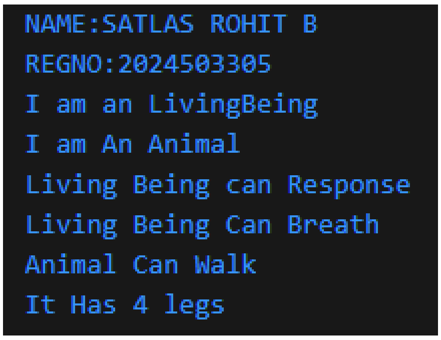
NAME: B Satlas Rohit

REGISTER NUMBER: 2024503305

6.1 Code

```
import java.util.Arrays;
package singleinheritance;
class LivingBeing{
    LivingBeing(){
        System.out.println("I am an LivingBeing");
    }
    void Breath(){
        System.out.println("Living Being Can Breath");
    }
    void response(){
        System.out.println("Living Being can Response");
    }
}
class Animal extends LivingBeing{
    Animal(){
        System.out.println("I am An Animal");
    }
    void Walk(){
        System.out.println("Animal Can Walk");
    }
    void NoOflegs(){
        System.out.println("It Has 4 legs");
    }

}
class ex1{
    public static void main(String[] args){
        System.out.println("NAME:SATLAS ROHIT B\nREGNO:2024503305");
        Animal ani=new Animal();
        ani.response();
        ani.Breath();
        ani.Walk();
        ani.NoOflegs();
    }
}
```

Output:

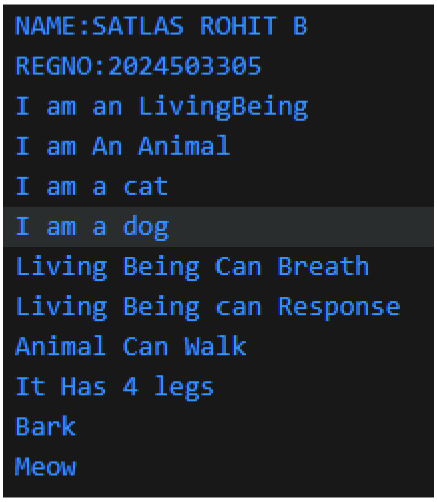


```
NAME:SATLAS ROHIT B
REGNO:2024503305
I am an LivingBeing
I am An Animal
Living Being can Response
Living Being Can Breath
Animal Can Walk
It Has 4 legs
```

6.2 Code

```
package Multileveinheritance;
class LivingBeing{
    LivingBeing(){
        System.out.println("I am an LivingBeing");
    }
    void Breath(){
        System.out.println("Living Being Can Breath");
    }
    void response(){
        System.out.println("Living Being can Response");
    }
}
class Animal extends LivingBeing{
    Animal(){
        System.out.println("I am An Animal");
    }
    void Walk(){
        System.out.println("Animal Can Walk");
    }
    void NoOflegs(){
        System.out.println("It Has 4 legs");
    }

}
class Cat extends Animal{
    Cat(){
        System.out.println("I am a cat");
    }
    void meow(){
        System.out.println("Meow");
    }
}
class Dog extends Cat{
    Dog(){
        System.out.println("I am a dog");
    }
    void Bark(){
        System.out.println("Bark");
    }
}
public class ex2 {
    public static void main(String[] args){
        System.out.println("NAME:SATLAS ROHIT B\nREGNO:2024503305");
        Dog d=new Dog();
        d.Breath();
        d.response();
        d.Walk();
        d.NoOflegs();
        d.Bark();
        d.meow();

    }
}
```

Output:



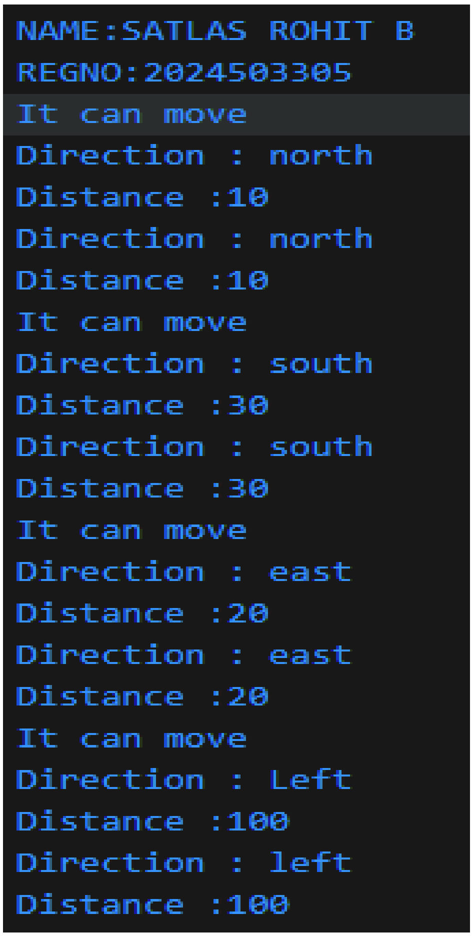
```
NAME:SATLAS ROHIT B
REGNO:2024503305
I am an LivingBeing
I am An Animal
I am a cat
I am a dog
Living Being Can Breath
Living Being can Response
Animal Can Walk
It Has 4 legs
Bark
Meow
```

6.3 Code

```
package polymorphism;
class Animal{
   void move(){
      System.out.println("It can move");
   }
   void move(String direction){
      System.out.println("Direction : "+direction);
   }
   void move(int distance){
      System.out.println("Distance :"+distance);
   }
   void move(String direction,int distance){
      System.out.println("Direction : "+direction);
      System.out.println("Distance :"+distance);
   }
}
class Dog extends Animal{
   void move(){
      System.out.println("It can move");
   }
   void move(String direction){
      System.out.println("Direction : "+direction);
   }
   void move(int distance){
      System.out.println("Distance :"+distance);
   }
   void move(String direction,int distance){
        System.out.println("Direction : "+direction);
        System.out.println("Distance :"+distance);
    }
}
class Cat extends Animal{
    void move(){
        System.out.println("It can move");
    }
    void move(String direction){
        System.out.println("Direction : "+direction);
    }
    void move(int distance){
        System.out.println("Distance :"+distance);
    }
    void move(String direction,int distance){
        System.out.println("Direction : "+direction);
        System.out.println("Distance :"+distance);
    }
}
class Bird extends Cat{
    void move(){
        System.out.println("It can move");
    }
    void move(String direction){
        System.out.println("Direction : "+direction);
    }
    void move(int distance){
        System.out.println("Distance :"+distance);
```

```
   }
   void move(String direction,int distance){
      System.out.println("Direction : "+direction);
      System.out.println("Distance :"+distance);
   }
}
public class ex3 {
   public static void main(String[] args){
      System.out.println("NAME:SATLAS ROHIT B\nREGNO:2024503305");
      Dog d=new Dog();
      Cat c=new Cat();
      Bird b=new Bird();
      Animal arrAnimal[]={d,c,b};
      String direction[]={"north","south","east"};
      int distance[]={10,30,20};
      for(int i=0;i<arrAnimal.length;i++){
        arrAnimal[i].move();
        arrAnimal[i].move(direction[i]);
        arrAnimal[i].move(distance[i]);
        arrAnimal[i].move(direction[i],distance[i]);
      }
      d.move();
      d.move("Left");
      d.move(100);
      d.move("left",100);
   }
}
```

Output:



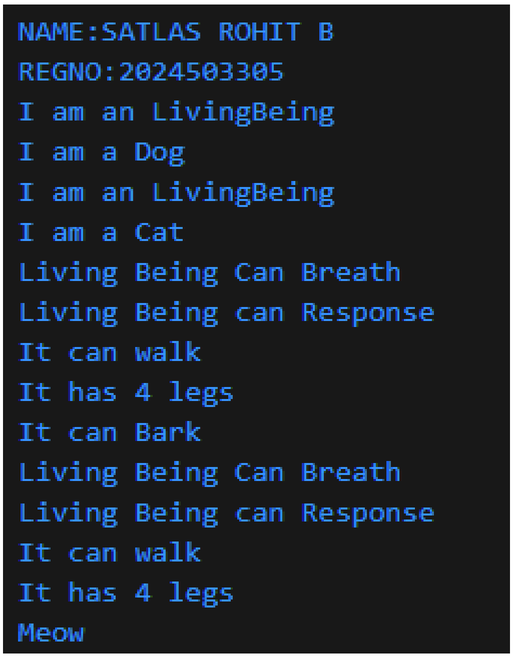
```
NAME:SATLAS ROHIT B
REGNO:2024503305
It can move
Direction : north
Distance :10
Direction : north
Distance :10
It can move
Direction : south
Distance :30
Direction : south
Distance :30
It can move
Direction : east
Distance :20
Direction : east
Distance :20
It can move
Direction : Left
Distance :100
Direction : left
Distance :100
```

6.4 Code

```
package hieraricalinheritance;
class LivingBeing{
  LivingBeing(String name){
    System.out.println("I am an LivingBeing");
  }
  void Breath(){
    System.out.println("Living Being Can Breath");
  }
  void response(){
    System.out.println("Living Being can Response");
  }
```

```
}
class LandAnimal extends LivingBeing{
    LandAnimal(String name){
        super(name);
    }
    void walk(){
        System.out.println("It can walk");
    }
    void numberofLegs(){
        System.out.println("It has 4 legs");
    }
}
class WaterAnimal extends LivingBeing{
    WaterAnimal(String name){
        super(name);
    }
    void swim(){
        System.out.println("It can swim");
    }
    void watertype(){
        System.out.println("Watertype is Salt Water");
    }
}
class Dog extends LandAnimal{
    Dog(String name){
        super(name);
        System.out.println("I am a "+name);
    }
    void bark(){
        System.out.println("It can Bark");
    }
}
class Cat extends LandAnimal{
    Cat(String name){
        super(name);
        System.out.println("I am a "+name);
    }
    void meow(){
        System.out.println("Meow");
    }
}
class ex4{
    public static void main(String[] args){
        System.out.println("NAME:SATLAS ROHIT B\nREGNO:2024503305");
        Dog d=new Dog("Dog");
        Cat c=new Cat("Cat");
        d.Breath();
        d.response();
        d.walk();
        d.numberofLegs();
        d.bark();
        c.Breath();
        c.response();
        c.walk();
        c.numberofLegs();
        c.meow();
    }
}
```

Output:



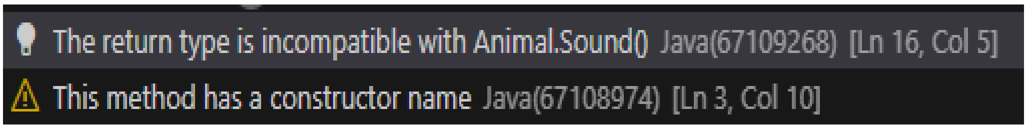
```
NAME:SATLAS ROHIT B
REGNO:2024503305
I am an LivingBeing
I am a Dog
I am an LivingBeing
I am a Cat
Living Being Can Breath
Living Being can Response
It can walk
It has 4 legs
It can Bark
Living Being Can Breath
Living Being can Response
It can walk
It has 4 legs
Meow
```

6.5

1.

```
package override;
class Animal{
    void Animal(){
        System.out.println("It is a Animal");
    }
    void Sound(){
        System.out.println("It makes Sound");
    }
}
class Dog extends Animal{
    Dog(){
        System.out.println("It is a Dog");
    }
    @Override
    int Sound(){
        System.out.println("It Makes sound1");
    }
}
public class ex5 {
    public static void main(String[] args){
        Animal ani=new Dog();
        ani.Sound();
    }
}
```

Output:

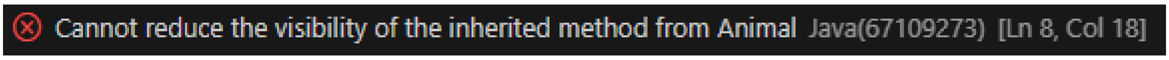


```
The return type is incompatible with Animal.Sound()  Java(67109268)  [Ln 16, Col 5]
⚠ This method has a constructor name  Java(67108974)  [Ln 3, Col 10]
```

2.

```
package override;
class Animal{
    public void Sound(){
        System.out.println("It makes Sound");
    }
}
class Dog extends Animal{
    private void Sound(){
        System.out.println("It Makes sound1");
    }
}
public class ex52 {
    public static void main(String[] args){
        Animal ani=new Dog();
        ani.Sound();
    }
}
```

Output:

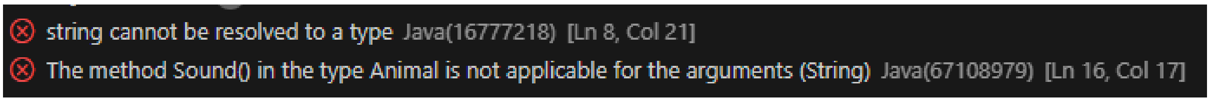


```
Cannot reduce the visibility of the inherited method from Animal  Java(67109273)  [Ln 8, Col 18]
```

3.

```
package override;
class Animal{
     void Sound(){
        System.out.println("It makes Sound");
    }
}
class Dog extends Animal{
     void  Sound(string sound){
        System.out.println("It Makes sound1"+sound);
    }
}


public class ex53 {
    public static void main(String[] args){
        Animal ani=new Dog();
        ani.Sound("bark");
    }
}
```

Output:

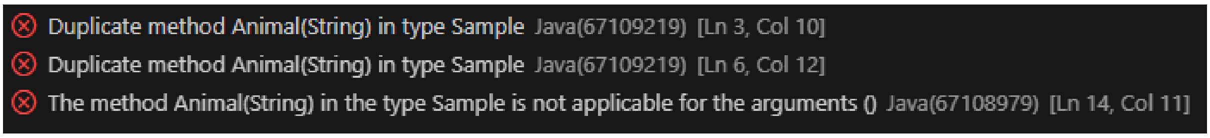


```
string cannot be resolved to a type  Java(16777218)  [Ln 8, Col 21]
The method Sound() in the type Animal is not applicable for the arguments (String)  Java(67108979)  [Ln 16, Col 17]
```

4.

```
package override;
class Sample{
   void Animal(String name){
      System.out.println("It is a Animal");
   }
   String Animal(String name){
      System.out.println("The animal is : "+name);
      return name;
   }
}
public class ex54 {
   public static void main(String[] args){
      Sample s=new Sample();
      s.Animal();
   }
}
```

Output:

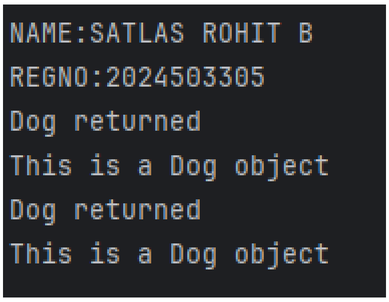


```
ⓧ Duplicate method Animal(String) in type Sample  Java(67109219)  [Ln 3, Col 10]
ⓧ Duplicate method Animal(String) in type Sample  Java(67109219)  [Ln 6, Col 12]
ⓧ The method Animal(String) in the type Sample is not applicable for the arguments ()  Java(67108979)  [Ln 14, Col 11]
```

5.

```
package upcastingdowncasting;
class Animal {
   Animal getAnimal() {
      System.out.println("Animal returned");
      return new Animal();
   }

   @Override
   public String toString() {
        return "This is an Animal object";
    }
}

class Dog extends Animal {
    @Override
    Dog getAnimal() {
        System.out.println("Dog returned");
        return new Dog();
    }

    @Override
    public String toString() {
        return "This is a Dog object";
    }
}

public class ex55 {
    public static void main(String[] args) {
        System.out.println("NAME:SATLAS ROHIT B\nREGNO:2024503305");
        Animal a = new Dog();

        Animal obj1 = a.getAnimal();
        System.out.println(obj1);

        Dog d = new Dog();
        Dog obj2 = d.getAnimal();
        System.out.println(obj2);
    }
}
```

Output:

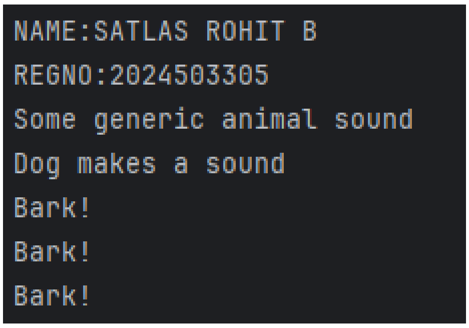


```
NAME:SATLAS ROHIT B
REGNO:2024503305
Dog returned
This is a Dog object
Dog returned
This is a Dog object
```

6.

```
package override;
class Animal {
    void sound() {
        System.out.println("Some generic animal sound");
    }

    void sound(String name) {
        System.out.println(name + " makes a sound");
    }
}

class Dog extends Animal {

    void sound(int times) {
        for (int i = 0; i < times; i++) {
            System.out.println("Bark!");
        }
    }
}

public class rightupcasting {
    public static void main(String[] args) {
        System.out.println("NAME:SATLAS ROHIT B\nREGNO:2024503305");
        Dog d = new Dog();

        d.sound();
        d.sound("Dog");
        d.sound(3);
    }
}
```

Output:

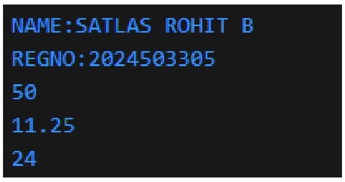


```
NAME:SATLAS ROHIT B
REGNO:2024503305
Some generic animal sound
Dog makes a sound
Bark!
Bark!
Bark!
```

7.

```
package override;
class MathUtil {
    static int multiply(int a, int b) {
        return a * b;
    }

    static double multiply(double a, double b) {
        return a * b;
    }

    static int multiply(int a, int b, int c) {
        return a * b * c;
    }
}
public class ex57 {
    public static void main(String[] args) {
        System.out.println("NAME:SATLAS ROHIT B\nREGNO:2024503305");
        System.out.println(MathUtil.multiply(5, 10));
        System.out.println(MathUtil.multiply(2.5, 4.5));
        System.out.println(MathUtil.multiply(2, 3, 4));
    }
}
```

Output:



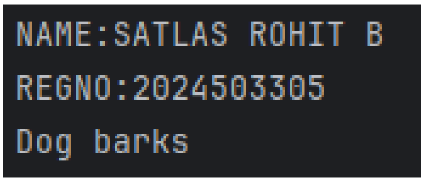
```
NAME:SATLAS ROHIT B
REGNO:2024503305
50
11.25
24
```

8.

Right Upcasting:

```
package upcastingdowncasting;
class Ani {
   void sound() { System.out.println("Animal sound"); }
}

class D extends Ani {
   void sound() { System.out.println("Dog barks"); }
   void fetch() { System.out.println("Dog fetches"); }
}



public class rightupcasting {
   public static void main(String[] args) {
      System.out.println("NAME:SATLAS ROHIT B\nREGNO:2024503305");
      Ani a = new D();
      a.sound();
   }
}
```

Output:



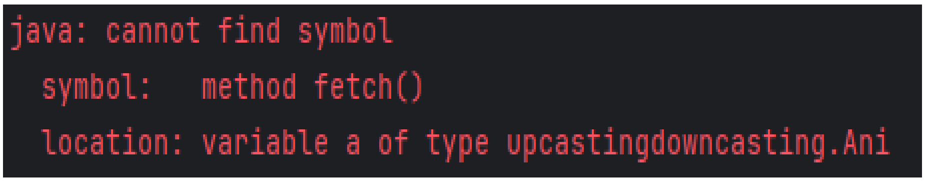
```
NAME:SATLAS ROHIT B
REGNO:2024503305
Dog barks
```

Wrong Upcasting:

```
package upcastingdowncasting;
class Ani {
   void sound() { System.out.println("Animal sound"); }
}

class D extends Ani {
   void sound() { System.out.println("Dog barks"); }
   void fetch() { System.out.println("Dog fetches"); }
}
```

```
public class rightupcasting {
    public static void main(String[] args) {
        System.out.println("NAME:SATLAS ROHIT B\nREGNO:2024503305");
        Ani a = new D();
        a.fetch();
    }
}
```

Output:

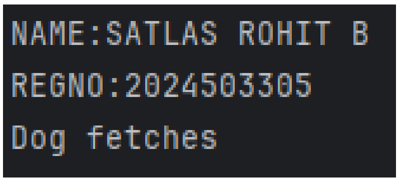


```
java: cannot find symbol
  symbol:   method fetch()
  location: variable a of type upcastingdowncasting.Ani
```

Right Downcasting:

```
package upcastingdowncasting;
class Ani {
    void sound() { System.out.println("Animal sound"); }
}
class D extends Ani {
    void sound() { System.out.println("Dog barks"); }
    void fetch() { System.out.println("Dog fetches"); }
}

public class rightupcasting {
    public static void main(String[] args) {
        System.out.println("NAME:SATLAS ROHIT B\nREGNO:2024503305");
        Ani a = new D();
        D d=(D) a;
        d.fetch();
    }
}
```

Output:

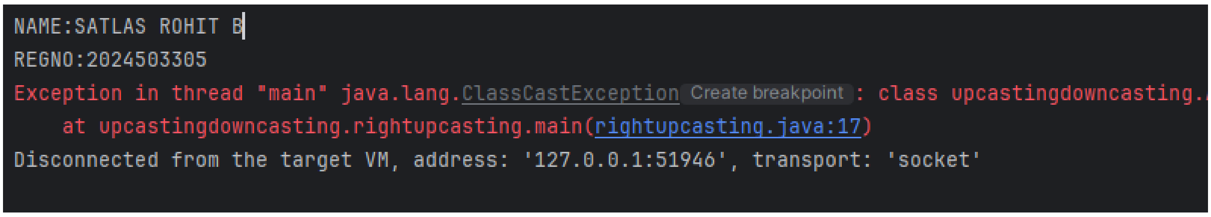


```
NAME:SATLAS ROHIT B
REGNO:2024503305
Dog fetches
```

Wrong Downcasting:

```
package upcastingdowncasting;
class Ani {
   void sound() { System.out.println("Animal sound"); }
}

class D extends Ani {
   void sound() { System.out.println("Dog barks"); }
   void fetch() { System.out.println("Dog fetches"); }
}



public class rightupcasting {
   public static void main(String[] args) {
      System.out.println("NAME:SATLAS ROHIT B\nREGNO:2024503305");
      Ani a = new Ani();
      D d=(D) a;
      d.sound();
   }
}
```

Output:



```
NAME:SATLAS ROHIT B
REGNO:2024503305
Exception in thread "main" java.lang.ClassCastException : class upcastingdowncasting.
    at upcastingdowncasting.rightupcasting.main(rightupcasting.java:17)
Disconnected from the target VM, address: '127.0.0.1:51946', transport: 'socket'
```