—:. LAB= 11:—

Q

Interface :—

```
public interface Pet?
    String name = null;
    String getname();
    public void setName( String name);
    void play ();
```

Abstract :—

```
abstract class Animal ?
    protected int legs;
    protected Animal (int legs) ?
        this. legs= legs; ?
    public void walk () ?
        Sout ("This Animal walks on + legs + legs "?;
    ?
    public abstract void eat();
?
```

Cat :—

```java
public class Cat extends Animal implements Pet {
    String name;
    public Cat(String name) {
        super(4);
        this.name = name;
    }
    public Cat() {
        this("Fluffy");
    }
    @Override
    public String getName() { return name; }

    @Override
    public void setName(String name) {
        this.name = name;
    }
    @Override
    public void play() { sout(name + " Likes to play with
                                String."); }

    public void eat() { sout("cats like to eat spiders
                                and fish."); }

}
```

Ayush ker. Jha
5000864406

=), Class:-

```java
public class Fish extends Animal implements Pet {
    private String name;
    @Override
    public String getName() { return name; }

    @Override
    public void setName(String name) { this.name = name; }

    public Fish {
        super(0);
        this.name = "fish";
    }

    public void play() { sout("Just keep swimming."); }
    public void eat() { System.out.("Fish eat pound scum"); }
    public void walk() {
        super.walk();
        System.out.println("Fish, of course, can't walk;
                    they swim.");
    }
}
```

Ayush kr. Jha
50008400

```java
public class PetMain {
    public static void playWithAnimal (Animal a) {
        if (a instance of Pet) {
            pet p = (pet )a;
            p. play ();
        } else {
            sout ("Danger! Wild Animal");
        }
    }

    // main

    public static void main (string [] args) {
        Cat C = new Cat ("Tom");
        C. eat();
        C. walk();
        C. play();
    }
}
```