1. package com.myproject.java;

```java
class Animal {
    void eat() {
        System.out.println("Animal is eating...");
    }
}

// Subclass 1 inheriting from Animal
class Dog extends Animal {
    void bark() {
        System.out.println("Dog is barking...");
    }
}

// Subclass 2 inheriting from Animal
class Cat extends Animal {
    void meow() {
        System.out.println("Cat is meowing...");
    }
}

// Subclass 3 inheriting from Dog and Cat
class Tiger extends Dog {
    void run() {
        System.out.println("Tiger is running...");
    }
}

// Subclass 4 inheriting from Cat
class Lion extends Cat {
    void roar() {
        System.out.println("Lion is roaring...");
    }
}

public class InheritanceExample {
    public static void main(String[] args) {
        Tiger tiger = new Tiger();
        tiger.eat();  // Inherited from Animal
        tiger.bark(); // Inherited from Dog
        tiger.run();  // Defined in Tiger class

        Lion lion = new Lion();
        lion.eat();   // Inherited from Animal
        lion.meow();  // Inherited from Cat
        lion.roar();  // Defined in Lion class
    }
```

```
}
```

2. package com.myproject.java;

```java
//Superclass Shape
abstract class Shape {
 abstract void draw();
}

//Subclass Square
class Square extends Shape {

 void draw() {
     System.out.println("Drawing a square");
 }
}

//Subclass Circle
class Circle extends Shape {

 void draw() {
     System.out.println("Drawing a circle");
 }
}

public class PolymorphismExample {
 public static void main(String[] args) {
     // Creating an array of Shape objects
     Shape[] shapes = new Shape[2];
     shapes[0] = new Square();
     shapes[1] = new Circle();

     // Polymorphic method calls
     for (Shape shape : shapes) {
         shape.draw();
     }
 }
}3. package com.myproject.java;

import java.util.Scanner;

class Organization {
    private int rating;

    // Constructor
    public Organization(int rating) {
        this.rating = rating;
    }

    public int getRating() {
        return rating;
    }


    public void setRating(int rating) {

        this.rating = rating;

    }

}


public class OrgRatingExample {

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);


        System.out.print("Enter the rating for the organization: ");

        int rating = scanner.nextInt();


        Organization org = new Organization(rating);


        System.out.println("Organization rating: " + org.getRating());




    }

}
```

4. package com.myproject.java;

```
import java.io.BufferedReader;

import java.io.IOException;
import java.io.InputStreamReader;


public class BufferedReaderExample {

    public static void main(String[] args) {

        BufferedReader reader = new BufferedReader(new
InputStreamReader(System.in));


        try {

            System.out.print("Enter some text: ");

            String input = reader.readLine();


            System.out.println("You entered: " + input);

        } catch (IOException e) { // handle any IOException that may
occur during the reading process

            e.printStackTrace();


        } finally {// close the BufferedReader object to release any
system resource

            try {

                reader.close();

            } catch (IOException e) {

                e.printStackTrace();

            }

        }

    }

}5.package com.myproject.java;
```

```
class ListNode {

    int data;

    ListNode next;


    public ListNode(int data) {

        this.data = data;

        this.next = null;

    }

}


class LinkedList {

    ListNode head;


    public LinkedList() {

        this.head = null;

    }


    public void insert(int data) { // inserts data

        ListNode newNode = new ListNode(data);


        if (head == null) {

            head = newNode;

        } else {

            ListNode current = head;
            while (current.next != null) { // keep on inserting data
as long as next is null
                current = current.next;
            }
            current.next = newNode;
        }
    }

    public void display() {   // display data
        ListNode current = head;

        if (current == null) {
            System.out.println("Linked list is empty.");
            return;
        }

        System.out.print("Linked list: ");
        while (current != null) {
            System.out.print(current.data + " ");
            current = current.next;
        }

    }
}
```

```
public class LinkedListExample {

    public static void main(String[] args) {

        LinkedList list = new LinkedList();



        list.insert(10);

        list.insert(20);

        list.insert(30);

        list.insert(40);



        list.display();

    }

}
```