**Code Examples**

**Example 1: Declaring an interface**

public interface **Animal**{

public static final int x = 4;

public void move();

}

**Example 2: Declaring an abstract class**

public abstract Bird {

public abstract void move();

public void eat() {

**System.out.println("The bird is eating");**

}

}

**Example 3: Declaring an abstract class in C++**

class bird {

public:

virtual void move() = 0;

virtual void eat() {

System.out.println("The bird is eating");

}

**};**

**Inheritance**

**Example 4: Inheriting from multiple Interface**

public class Chicken implements Animal, X {

public void move(){

System.out.println("The chicken is walking");

}

}

**Example 5: Inheriting from Abstract class and multiple interfaces**

public class Chicken extends Bird implements X,Y {

public void move(){

System.out.println("The chicken is walking");

}

Bird b = new Chicken();

b.eat();

}

**Exercise**

**Exercise 1:**

public xxxxxx Bird {

int count = 3 ; (1)

protected String name; (2)

private int x = 7; (3)

public static int totals; (4)

**public Birds**(String name){ (5)

this.name = name;

}

**public abstract void move();** (6)

**public void eat();**  (7)

**public:**

virtual void f() = 0; (8)

}

}

**Table1**: Fill in the class members from exercise 1 that are found in each of the components in the table.

|  |  |  |
| --- | --- | --- |
| **Interface** | **Abstract Class** | **C ++ Code** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |