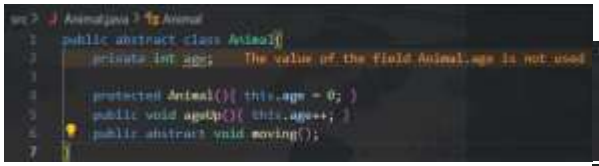
Zid'Avwa Al Bari'i

244107020083 / TI – 2I / 26

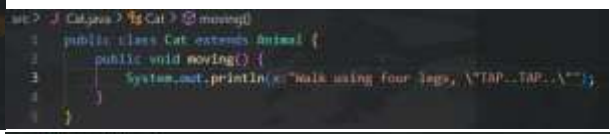
Jobsheet 10

Exercise 1

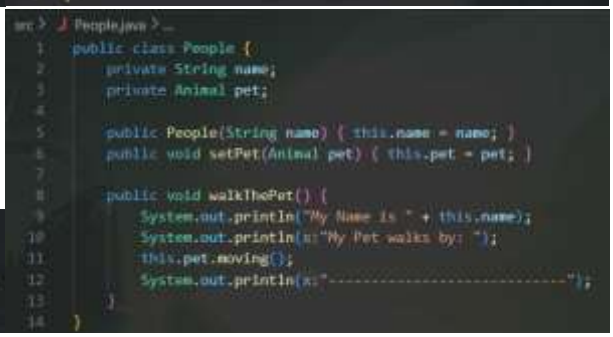
Codes



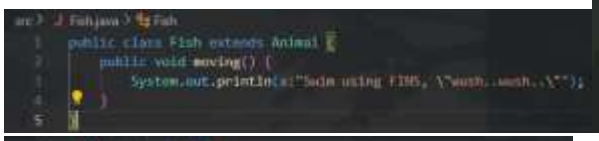
```java
public abstract class Animal{
    private int age;    The value of the field Animal.age is not used

    protected Animal(){ this.age = 0; }
    public void ageUp(){ this.age++; }
    public abstract void moving();
}
```



```java
public class Cat extends Animal {
    public void moving() {
        System.out.println("Walk using four legs, \"TAP..TAP..\"");
    }
}
```



```java
public class People {
    private String name;
    private Animal pet;

    public People(String name) { this.name = name; }
    public void setPet(Animal pet) { this.pet = pet; }

    public void walkThePet() {
        System.out.println("My Name is " + this.name);
        System.out.println("My Pet walks by: ");
        this.pet.moving();
        System.out.println("--------------------------");
    }
}
```



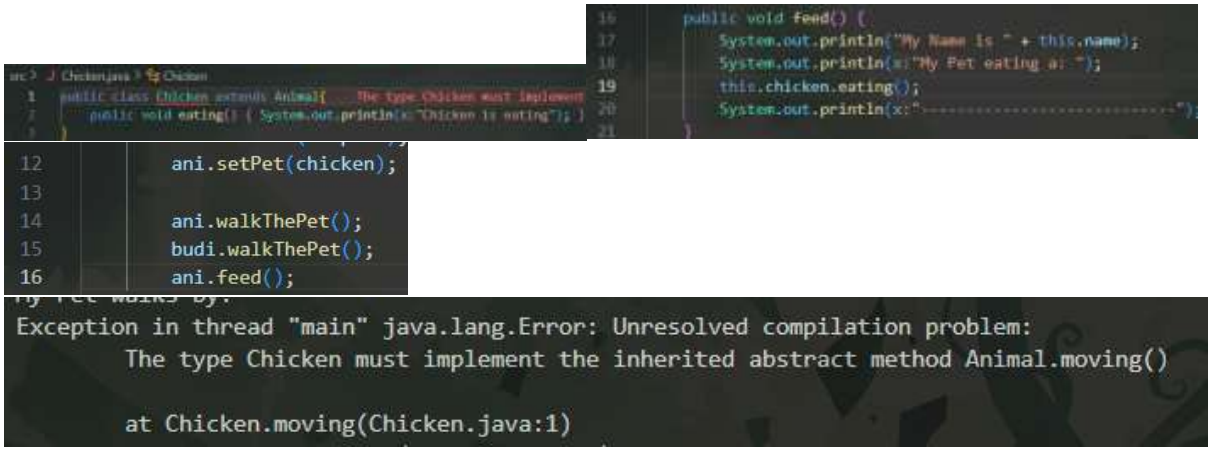
```java
public class Fish extends Animal {
    public void moving() {
        System.out.println("Swim using FINS, \"wush..wush..\"");
    }
}
```

```java
public class Main {
    Run | Debug
    public static void main(String[] args) {
        Cat wildCat = new Cat();
        Fish dolphin = new Fish();

        People ani = new People(name:"Ani");
        People budi = new People(name:"Budi");

        ani.setPet(wildCat);
        budi.setPet(dolphin);

        ani.walkThePet();
        budi.walkThePet();
    }
}
```

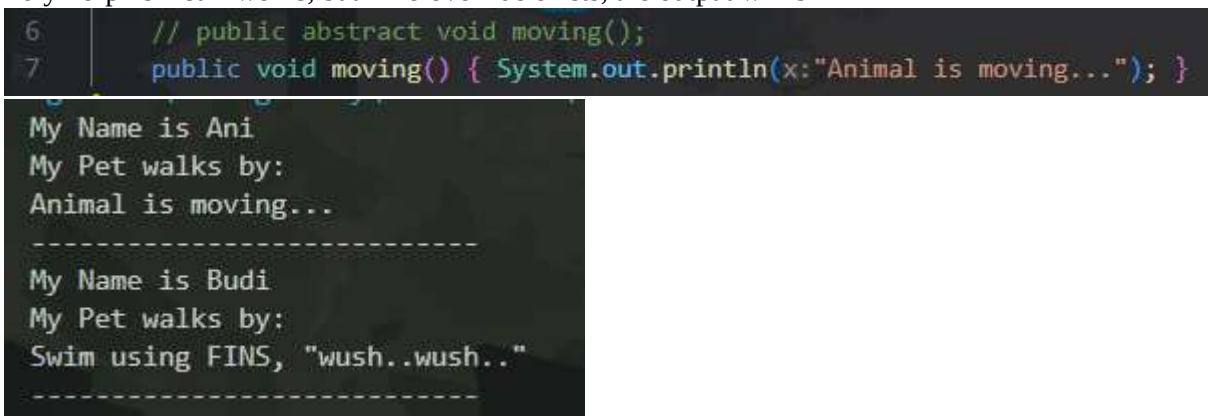
Questions & Answers:

Discussion: Is it permissible if a class that extends an abstract class does not implement the abstract method in its parent class? Prove!

No, it's not permissible, unless the subclass is also declared abstract. If a subclass doesn't implement the abstract method, Java forces it to become abstract too.



1. **Provide a related explanation of the program above**
   The program demonstrates inheritance and polymorphism.
   Animal defines an abstract method moving().
   Cat and Fish implement this method differently.
   People holds an Animal reference and calls pet.moving() — showing dynamic behavior depending on the object's actual type (Cat or Fish).
2. **Show the compilation results of the program and give a brief explanation if the moving method () is changed to method abstract!**
   If moving() in Animal is no longer abstract (given a body), the program still compiles and runs.
   Subclasses can override it or use the default version.
   Polymorphism still works, but if no override exists, the output will be from Animal's method.



3. **Show the results of the program compilation and provide a brief explanation if there is no overriding of the moving method()**
   If Cat and Fish remove their moving() methods while Animal keeps it abstract, compilation fails because abstract methods must be implemented.
   To fix it, Cat and Fish would also need to be declared abstract. Will show the same error as the Chicken Class from the discussion.
4. **Show the results of the program compilation and provide a brief explanation if the abstract method move() declared in the Fish Class**
   If Fish declares moving() as abstract, then Fish must also be abstract and cannot be instantiated.
   Main would fail at new Fish() because abstract classes can't be created directly, a concrete subclass of Fish must implement moving() first.