

Department of Computer Science
Final Term Exam Spring 2020

NAME	WASEEM KHAN
ID NO	14306
DEPARTMENT	BS (CS 5 th)
Instructor:	M.Ayub Khan
Subject:	Object Oriented Programming

Note:

At the top of the answer sheet there must be the ID, Name and semester of the concerned Student.

Students must have to provide the output of their respective programs. Students have same answers or programs will be considered fail. Programs in Java or codes should be explained clearly.

As this paper is online so incase of any ambiguity my Whatsapp no. is 034499121116.

**Each question carry equal marks.
Please answer briefly.**

=====

Q1. a. Why access modifiers are used in java, explain in detail Private and Default access modifiers?

=====

ANS . ACCESS MODIFIERDS

=====

In must have seen have seen public , private and protected keywords while practicing java programs , there are called access modifiers . an access modifier resist the access modifier restricts the access the class , constructor , data member and method in another class .. In java we have four access modifiers,

1. Default access modifiers

=====

2. Private access modifiers

=====

3. Protected access modifiers

=====

4. Public access modifiers

=====

OR

=====

The access modifiers in Java specifies the accessibility or scope of a field, method, constructor, or class. We can change the access level of fields, constructors, methods, and class by applying the access modifier on it.

1. Default access modifier

=====

When we do not mention any access modifier, it is called default access modifier. The scope of this modifier is limited to the package only. This means that if we have a class with the default access modifier in a package, only those classes that are in this package can access this class. No other class outside this package can access this class. Similarly, if we have a default method or data member in a class, it would not be visible in the class of another package.

The access level of a default modifier is only within the package. It cannot be accessed from outside the package. If you do not specify any access level, it will be the default.

EXAMPLE IN DEFAULT ACCESS MODIFIER

=====

in this example we have two classes, Test class is trying to access the default method of Addition class, since class Test belongs to a different package, this program would throw compilation error, because the scope of default modifier is limited to the same package in which it is declared.

```
package abcpackage;

public class Addition {
    /* Since we didn't mention any access modifier here, it would
     * be considered as default.
     */
    int addTwoNumbers(int a, int b){
        return a+b;
    }
}
```

Test.java

```
package xyzpackage;

/* We are importing the abcpackage
 * but still we will get error because the
 * class we are trying to use has default access
 * modifier.
 */
import abcpackage.*;

public class Test {
    public static void main(String args[]){
        Addition obj = new Addition();
        /* It will throw error because we are trying to access
         * the default method in another package
         */
        obj.addTwoNumbers(10, 21);
    }
}
```

Output:

```
Exception in thread "main" java.lang.Error: Unresolved compilation problem:
The method addTwoNumbers(int, int) from the type Addition is not visible
at xyzpackage.Test.main(Test.java:12)
```

2. Private access modifier

=====

The access level of a private modifier is only within the class. It cannot be accessed from outside the class

The scope of private modifier is limited to the class only.

1. Private Data members and methods are only accessible within the class .
2. Class and Interface cannot be declared as private .
3. If a class has private constructor then you cannot create the object of that class from outside of the class.

EXAMPLE IN PRIVATE ACCESS MODIFIERS

=====

This example throws compilation error because we are trying to access the private data member and method of class ABC in the class Example. The private data member and method are only accessible within the class.

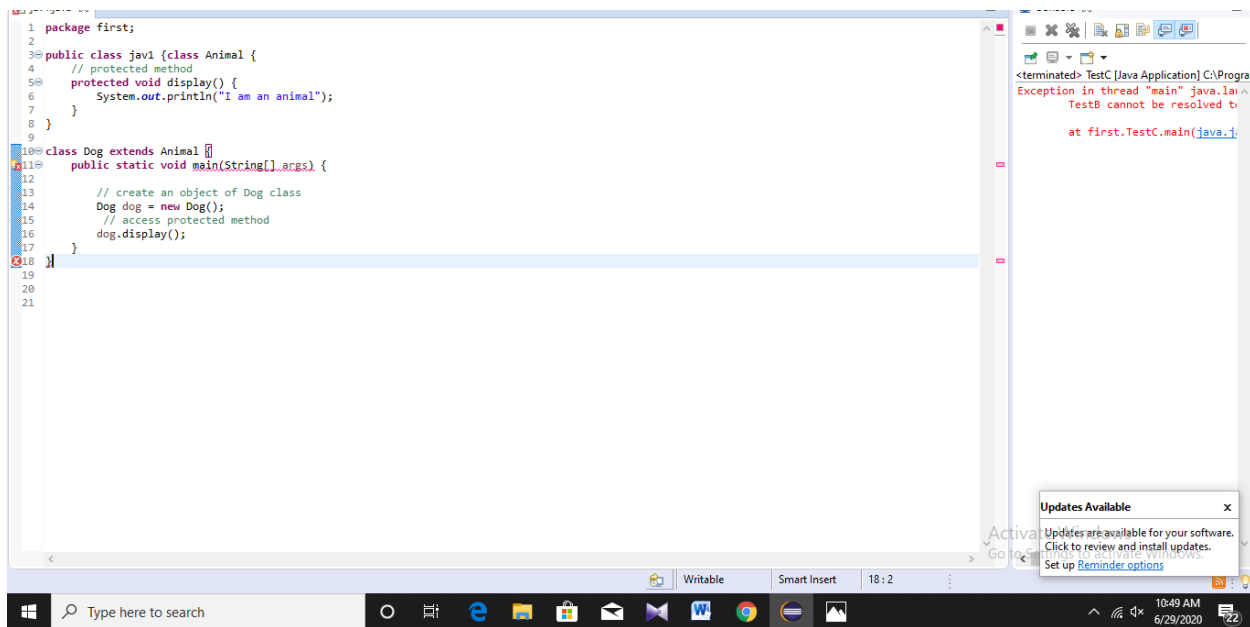
```
class ABC{
    private double num = 100;
    private int square(int a){
        return a*a;
    }
}
public class Example{
    public static void main(String args[]){
        ABC obj = new ABC();
        System.out.println(obj.num);
        System.out.println(obj.square(10));
    }
}
```

Output:

```
Compile - time error
```

b. Write a specific program of the above mentioned access modifiers in java.

Program : a specific program of the anove mentioned access modifiers in java



```
1 package first;
2
3 public class jav1 {class Animal {
4     // protected method
5     protected void display() {
6         System.out.println("I am an animal");
7     }
8 }
9
10 class Dog extends Animal {
11     public static void main(String[] args) {
12
13         // create an object of Dog class
14         Dog dog = new Dog();
15         // access protected method
16         dog.display();
17     }
18 }
19
20
21
```

<terminated> TestC [Java Application] C:\Program...
Exception in thread "main" java.lai...
TestB cannot be resolved to...
at first.TestC.main(java.i...

Updates Available
Updates are available for your software.
Click to review and install updates.
Set up Reminder options

a specific program of the anove mentioned access modifiers in java

```
public class MyClass {
    final int x = 10;
    final double PI = 3.14;

    public static void main(String[] args) {
        MyClass myObj = new MyClass();
        myObj.x = 50; // will generate an error
        myObj.PI = 25; // will generate an error
        System.out.println(myObj.x);
    }
}
```

```
MyClass.java:7: error: cannot assign a value to final variable x
    myObj.x = 50;
            ^
MyClass.java:8: error: cannot assign a value to final variable PI
    myObj.PI = 25;
            ^
2 errors
```

Waiting for pixel.advertising.com...
Type here to search
Activate Windows
Go to Settings

=====

Q2. a. Explain in detail Public and Protected access modifier .

=====

ANS

=====

PROTECTED ACCESS MODIFIERS

=====

Variables, methods, and constructors, which are declared protected in a superclass can be accessed only by the subclasses in other package or any class within the package of the protected members class. The access level of a protected modifier is within the package and outside the package through child class. If you do not make the child class, it cannot be accessed from outside the package .

Example

=====

In this example, we have created the two packages pack and mypack. The A class of pack package is public, so can be accessed from outside the package. But msg method of this package is declared as protected, so it can be accessed from outside the class only through inheritance.

```
//save by A.java
package pack;
public class A{
protected void msg(){System.out.println("Hello");}
}
```

```
//save by B.java
package mypack;
import pack.*;

class B extends A{
public static void main(String args[]){
B obj = new B();
obj.msg();
}
}
```

Output:Hello

Public access modifier

=====

The members, methods and classes that are declared public can be accessed from anywhere. This modifier doesn't put any restriction on the access . or The access level of a public modifier is everywhere. It can be accessed from within the class, outside the class, within the package and outside the packages .

Example public access modifier

=====

public modifier and class Test is able to access this method without even extending the Addition class. This is because public modifier has visibility everywhere.

```
package abcpackage;  
  
public class Addition {  
  
    public int addTwoNumbers(int a, int b){  
        return a+b;  
    }  
}
```

Test.java

```
package xyzpackage;  
import abcpackage.*;  
class Test{  
    public static void main(String args[]){  
        Addition obj = new Addition();  
        System.out.println(obj.addTwoNumbers(100, 1));  
    }  
}
```

Output:

```
101
```

=====
b. Write a specific program of the above mentioned access modifiers in java.
=====

Programs

=====

Program in above mentioned access modification in java

```
package com.journaldev.access;

import com.journaldev.access.TestA;

public class TestB {

    public static void main(String args[]) {
        new TestA().methodPublic();
        new TestA().methodProtected();
        new TestA().methodDefault();
    }

    public void methodPublic() {
    }

    protected void methodProtected() {
    }

    void methodDefault() {
    }

    private void methodPrivate() {
    }
}
```

Activate Windows
Go to Settings to activate Windows.



Program in above mentioned access modification in java

```
class MyClass {
    public static void main(String[] args) {
        // create an object of the Student class (which inherits attributes and methods from Person)
        Student myObj = new Student();

        System.out.println("Name: " + myObj.fname);
        System.out.println("Age: " + myObj.age);
        System.out.println("Graduation Year: " + myObj.graduationYear);
        myObj.study(); // call abstract method
    }
}
```

```
Name: John
Age: 24
Graduation Year: 2018
Studying all day long
```

Activate Windows
Go to Settings to activate Windows.



=====
Q3. a. What is inheritance and why it is used, discuss in detail ?
=====

ANS

INHERITANCE

=====

Inheritance is a mechanism in which one class acquires the property of another class.

Or

Inheritance is a mechanism in which one class acquires the property of another class. For example, a child inherits the traits of his/her parents. With inheritance, we can reuse the fields and methods of the existing class. Hence, inheritance facilitates Reusability and is an important concept of OOPs.

=====

Why we use inheritance

=====

inheritance is used when a class wants to use/inherit the features of another existing class. The class that wants to use the feature of another class, is called subclass, whereas the class whose features are to be used is referred to as superclass. Inheritance is used to use the existing features of a class.

EXAMPLE

=====

For example, a child inherits the traits of his/her parents. With inheritance, we can reuse the fields and methods of the existing class. Hence, inheritance facilitates Reusability and is an important concept of OOPs

Types of Inheritance

=====

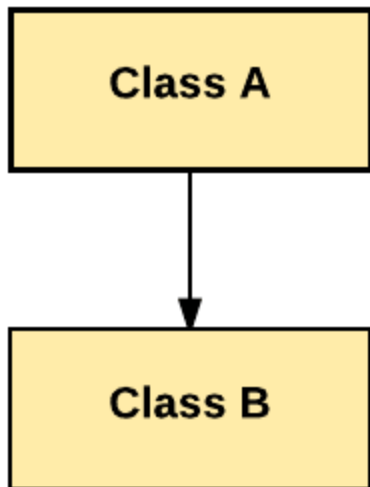
There are five types of inheritance

=====

1 Single Inheritance

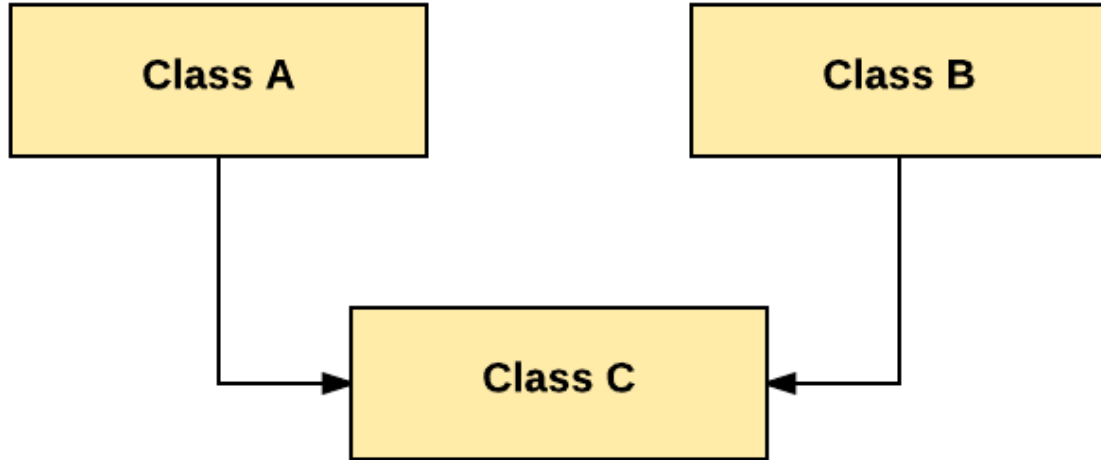
=====

In Single Inheritance one class extends another class (one class only).



2 Multiple Inheritance

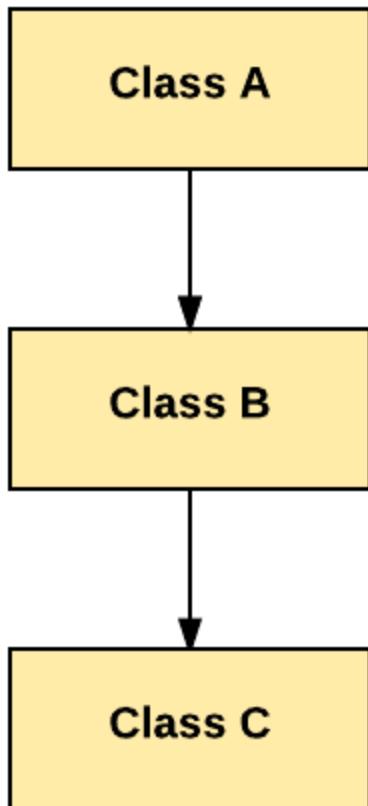
In Multiple Inheritance, one class extending more than one class. Java does not support multiple inheritance.



3 Multilevel Inheritance

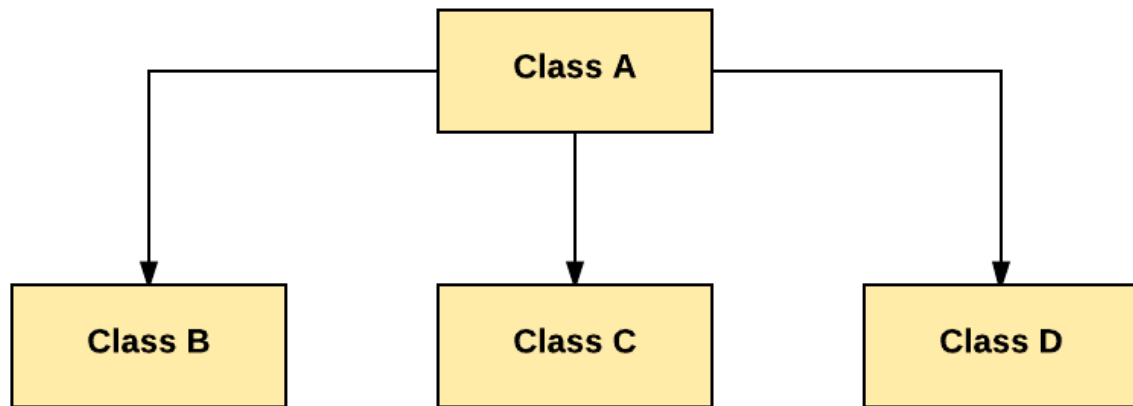
=====

In Multilevel Inheritance, one class can inherit from a derived class. Hence, the derived class becomes the base class for the new class.



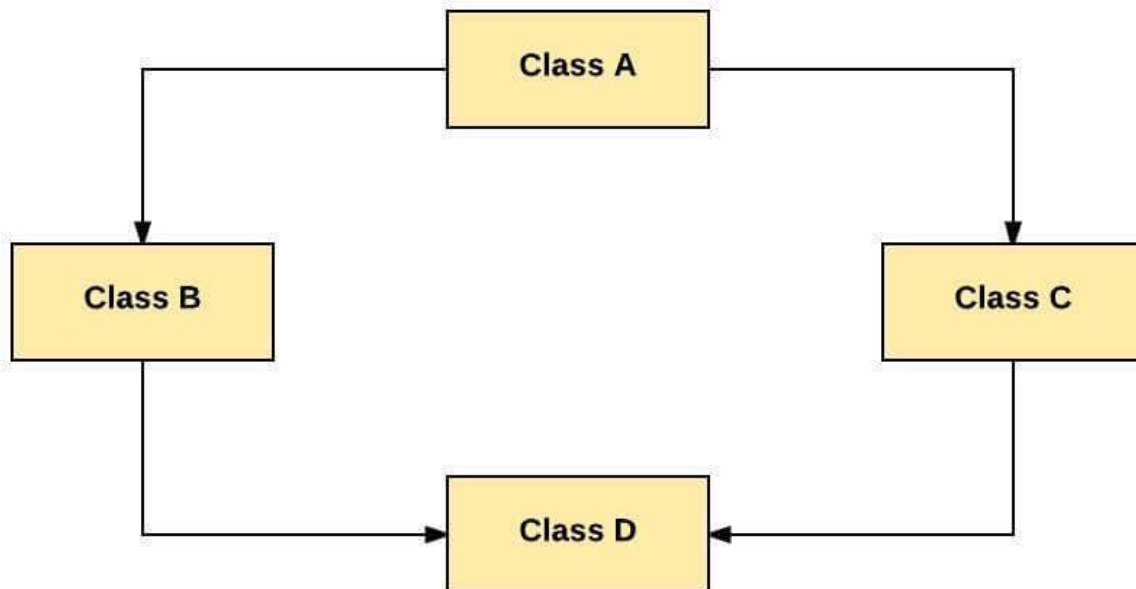
4 Hierarchical Inheritance

In Hierarchical Inheritance, one class is inherited by many sub classes.



5 Hybrid Inheritance

Hybrid inheritance is a combination of Single and Multiple inheritance.



b. Write a program using Inheritance class on Animal in java.

=====
Program: program using inheritance class animal in java .
=====

```
class Animal {  
    public void animalSound() {  
        System.out.println("The animal makes a sound");  
    }  
}  
  
class Pig extends Animal {  
    public void animalSound() {  
        System.out.println("The pig says: wee wee");  
    }  
}  
  
class Dog extends Animal {  
    public void animalSound() {  
        System.out.println("The dog says: bow wow");  
    }  
}  
  
class MyMainClass {  
    public static void main(String[] args) {  
        Animal myAnimal = new Animal();  
        Animal myPig = new Pig();  
        Animal myDog = new Dog();  
  
        myAnimal.animalSound();  
        myPig.animalSound();  
        myDog.animalSound();  
    }  
}
```

```
The animal makes a sound  
The pig says: wee wee  
The dog says: bow wow
```

Activate Windows
Go to Settings to activate W

=====
Q4. a. What is polymorphism and why it is used, discuss in detail ?
=====

ANS
Polymorphism

=====

In object-oriented programming, polymorphism (from the Greek meaning "having multiple forms") is the characteristic of being able to assign a different meaning or usage to something in different contexts - specifically, to allow an entity such as a variable, a function, or an object to have more than one form.

Or In other words, polymorphism allows you to define one interface and have multiple implementations.

Why we used polymorphism

=====

Polymorphism is the ability of any data to be processed in more than one form. ... Polymorphism is one of the most important concept of object oriented programming language. The most common use of polymorphism in object-oriented programming occurs when a parent class reference is used to refer to a child class object.

Example

=====

```
public interface Vegetarian{}  
public class Animal{}  
public class Deer extends Animal implements Vegetarian{}
```

Now, the Deer class is considered to be polymorphic since this has multiple inheritance. Following are true for the above examples

A Deer IS-A Animal

A Deer IS-A Vegetarian

A Deer IS-A Deer

A Deer IS-A Object

When we apply the reference variable facts to a Deer object reference, the following declarations are legal

b. Write a program using polymorphism in a class on Employee in java.

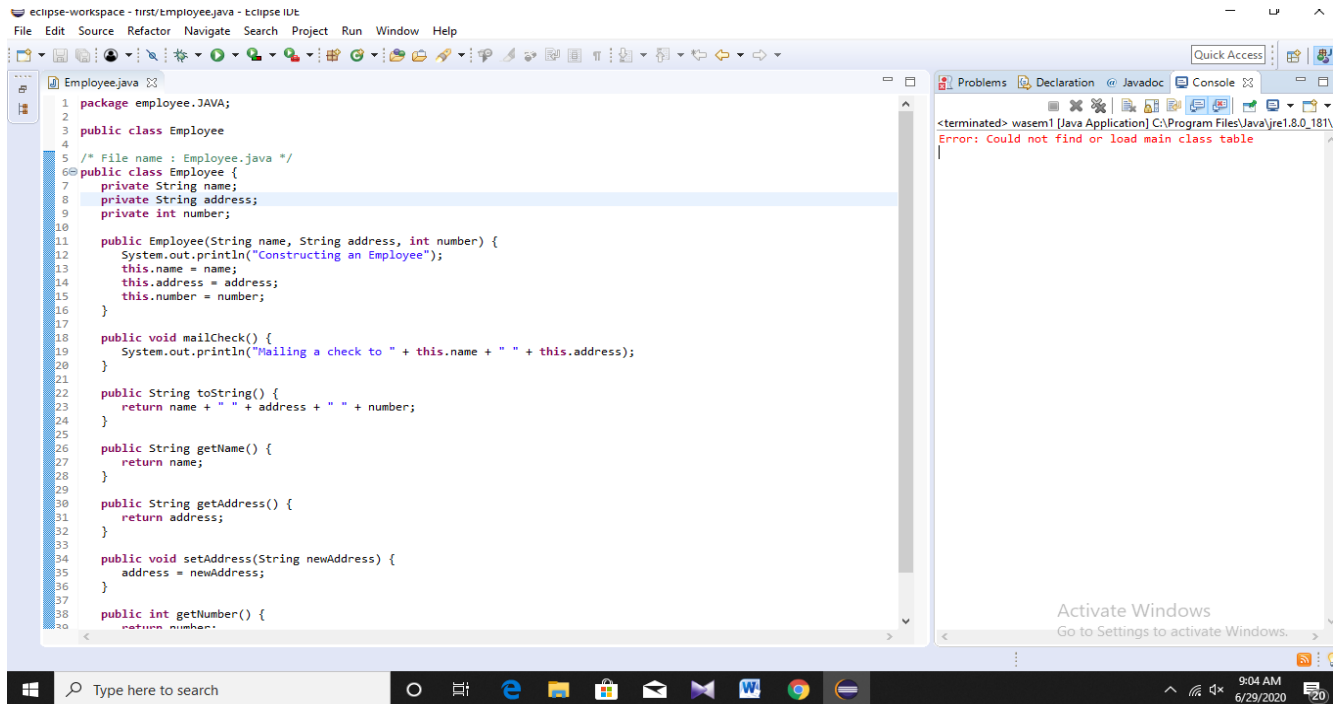
=====

ANS

=====

PAROGRAMME

=====



=====

Q5. a. Why abstraction is used in OOP, discuss in detail ?

=====

ANS

Abstraction in OOP

=====

Abstraction is selecting data from a larger pool to show only the relevant details of the object to the user. Abstraction “shows” only the essential attributes and “hides” unnecessary information. It helps to reduce programming complexity and effort. It is one of the most important concepts of OOPs.

Example

=====

suppose you want to create a banking application and you are asked to collect all the information about your customer. There are chances that you will come up with following information about the customer ,

1 full name
2 address
3 contact nmb
4 tax information
✓ 5 favorite food
✓ 6 favorite works
✓ 7 favorite movie
✓ 8 favorite book

Ok ,we might not need the all need customer information for a banking application .

But, not all of the above information is required to create a banking application.

so, you need to select only the useful information for your banking application from that pool. Data like name, address, tax information, etc. make sense for a banking application .

✓ full name
✓ tax information
✓ address
✓ connect number

Since we have fetched/removed/selected the customer information from a larger pool, the process is referred as Abstraction. However, the same information once extracted can be used for a wide range of applications. For instance, you can use the same data for hospital application, job portal application,

Abstraction in Java

=====

Abstraction in JAVA “shows” only the essential attributes and “hides” unnecessary details of the object from the user. In Java, abstraction is accomplished using Abstract classes, Abstract methods, and interfaces. Abstraction helps in reducing programming complexity and effort.

Abstract Class

=====

A class which is declared “abstract” is called as an abstract class. It can have abstract methods as well as concrete methods. A normal class cannot have abstract methods.

Abstract Method

=====

A method without a body is known as an Abstract Method. It must be declared in an abstract class. The abstract method will never be final because the abstract class must implement all the abstract methods.

=====

b. Write a program on abstraction in java.

=====

Program

=====

Program on abstraction in java

=====

```
// Abstract class
abstract class Animal {
    // Abstract method (does not have a body)
    public abstract void animalSound();
    // Regular method
    public void sleep() {
        System.out.println("Zzz");
    }
}

// Subclass (inherit from Animal)
class Pig extends Animal {
    public void animalSound() {
        // The body of animalSound() is provided here
        System.out.println("The pig says: wee wee");
    }
}

class MyMainClass {
    public static void main(String[] args) {
        Pig myPig = new Pig(); // Create a Pig object
        myPig.animalSound();
        myPig.sleep();
    }
}
```

```
The pig says: wee wee
Zzz
```

Activate Windows
Go to Settings to activate Windows.

