

Department of Computer Science
Final Term Exam Spring 2020

Subject: Object Oriented Programming

BS (CS, SE)

Student name: Wajid ullah

ID:12995

Section: A

Instructor: M.Ayub Khan

There are total 5 questions in this paper.

Max Marks: 50

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 in case of any ambiguity my WhatsApp no. is 034499121116.

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

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

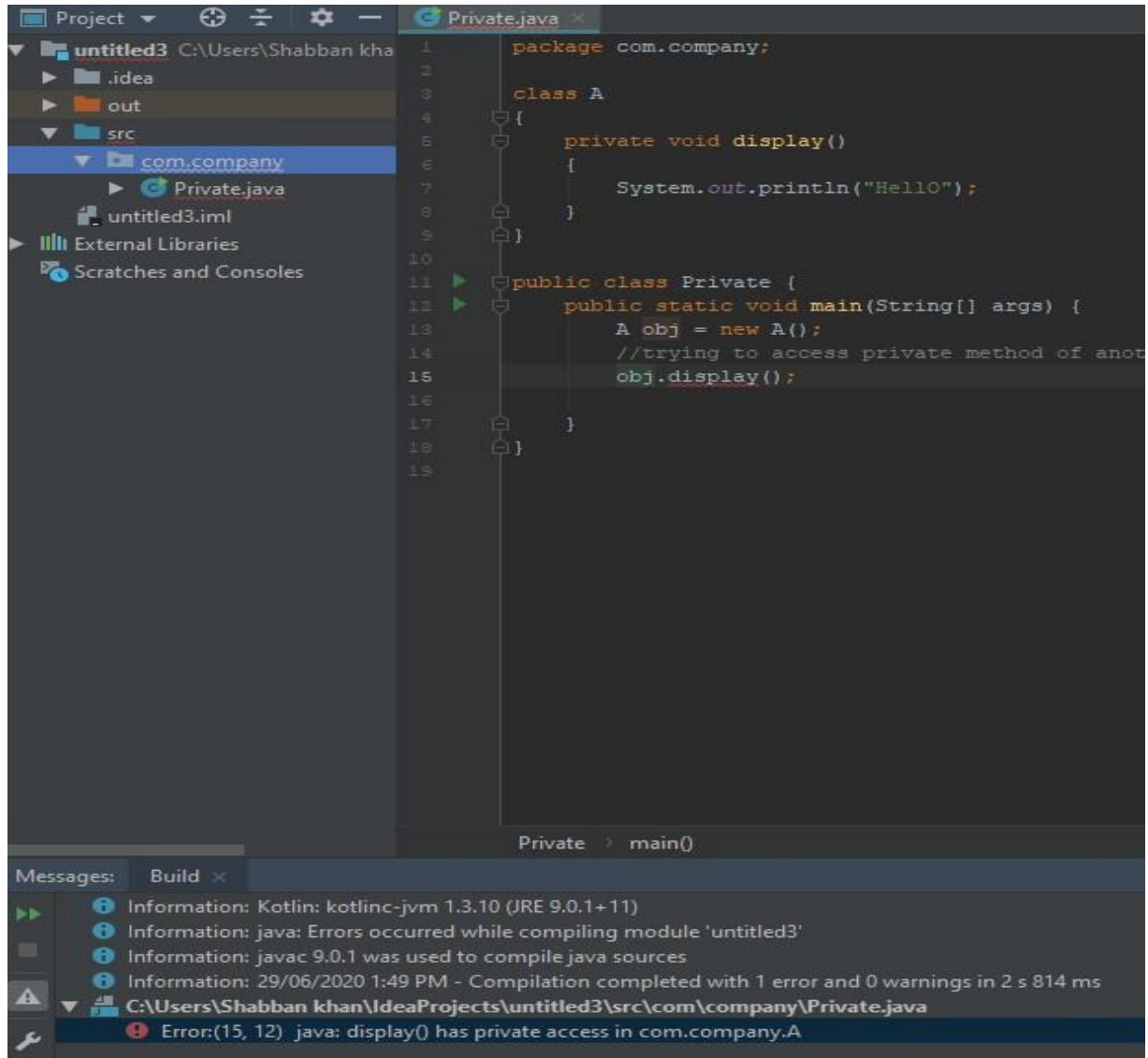
Ans: Access modifiers provides abstraction to the class in java we use it for different purposes because there are different access modifiers each one with its own specialty.

Private Access modifiers: If we declare any member as private its scope will be limited within class only in which it is declared. A class can't be private except the nested class.

Default Access modifiers: if we don't use any modifier with the members, java will treat them as default by default. Default members can be accessed only within a package only.

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

Private:



The screenshot shows an IDE window with a project named 'untitled3'. The file explorer on the left shows the project structure, including a package 'com.company' containing 'Private.java'. The main editor displays the following Java code:

```
1 package com.company;
2
3 class A
4 {
5     private void display()
6     {
7         System.out.println("Hello");
8     }
9 }
10
11 public class Private {
12     public static void main(String[] args) {
13         A obj = new A();
14         //trying to access private method of another class
15         obj.display();
16     }
17 }
18
19 }
```

The IDE's Messages window at the bottom shows the following output:

```
Build x
Information: Kotlin: kotlinc-jvm 1.3.10 (JRE 9.0.1+11)
Information: java: Errors occurred while compiling module 'untitled3'
Information: javac 9.0.1 was used to compile java sources
Information: 29/06/2020 1:49 PM - Compilation completed with 1 error and 0 warnings in 2 s 814 ms
C:\Users\Shabban khan\IdeaProjects\untitled3\src\com\company\Private.java
Error:(15, 12) java: display() has private access in com.company.A
```

Default:

```
package com.company;

public class Default {
    public static void main(String[] args) {

        OtherClass otherClass= new OtherClass();
        otherClass.display();
    }
}
```

```
package com.company;

public class OtherClass {

    void display()
    {
        System.out.println("Hello World!");
    }
}
```

```
"C:\Program Files\Java\jdk-9.0.1\bin\java.exe" -Djava.class.path=. Default.class
Hello World!
```

```
Process finished with exit code 0
```

Q2. a. Explain in detail Public and Protected access modifiers?

Public Access modifiers: It means visible everywhere.

if we declare any member as public we can access it everywhere. fields, methods, blocks declared inside a public class can be accessed from any class belonging to the java Universe.

Protected Access modifiers: Protected member can be accessible only within the package but outside the package through the inheritance only. A class can't be protected except the nested class.

Both are explained in the following code example:

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

Protected and Public:

```
package com.company;

public class Protec extends OtherClass {
    public static void main(String[] args) {
        Protec protec=new Protec();
        protec.display();
    }
}
```

```
package com.company;

public class OtherClass {
    protected void display()
    {
        System.out.println("Hello World! Protected");
    }
}
```

```
"C:\Program Files\Java\jdk-9.0.1\bin\java.exe"
Hello World! Protected

Process finished with exit code 0
```

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

Ans: Inheritance in Java is a mechanism in which one object acquires all the properties and behaviors of a parent object. The purpose behind inheritance in Java is that you can create new classes that are built upon existing classes. Why we use it in java because of two major benefits:

Code reusability

Method overriding

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

```
package com.company;

import java.util.Scanner;

public class Main {

    public static void main(String[] args) {
        Scanner cin= new Scanner(System.in);
        System.out.println("1. Bird\n2. Turtle\n3. Shepherd\n4. Huskey");
        System.out.println("Please Enter a Number");
        int choice= cin.nextInt();
        bird Bird= new bird( color: "golden", legs: 2);
        turtle Turtle = new turtle( color: "green", legs: 4);
        Shepherd shepherd= new Shepherd( color: "brown", legs: 4);
        Huskey huskey= new Huskey( color: "white", legs: 4);

        switch (choice){
            case 1:
                Bird.movement();
                Bird.getDetails();
                break;
            case 2:
                Turtle.movement();
                Turtle.getDetails();
                break;
            case 3:
                shepherd.movement();
                shepherd.getDetails();
                break;
            case 4:
                huskey.movement();
                huskey.getDetails();
                break;
            default:
                System.out.println(" Entered wrong Number ");
                break;
        }

        // write your code here
    }
}
```

```
package com.company;

public class Animal {
    private String color;
    private int legs;

    public Animal(String color, int legs) {
        this.color = color;
        this.legs = legs;
    }

    public void movement(){
        System.out.println("movement begins ");
    }

    public void getDetails(){
        System.out.println("Legs: "+this.legs );
        System.out.println("Color: "+this.color );
    }
}
```

```
package com.company;

public class turtle extends Animal {

    public turtle(String color, int legs) {
        super(color, legs);
    }

    @Override
    public void movement() {
        super.movement();
        System.out.println(" waking ");
    }
}
```

```
package com.company;

public class Dog extends Animal{
    public Dog(String color, int legs) {
        super(color, legs);
    }

    @Override
    public void movement() {
        super.movement();
    }
}
```

```
package com.company;

public class Shepherd extends Dog {
    public Shepherd(String color, int legs) {
        super(color, legs);
    }

    @Override
    public void movement() {
        super.movement();
        System.out.println(" running ");
    }
}
```



```
package com.company;

public class Huskey extends Dog {
    public Huskey(String color, int legs) {
        super(color, legs);
    }

    @Override
    public void movement() {
        super.movement();
        System.out.println(" Running ");
    }
}
```

```
1. Bird
2. Turtle
3. Shepherd
4. Haskey
Please Enter a Number
2
movement begins
waking
Legs: 4
Color: green

Process finished with exit code 0
```

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

Polymorphism is defined as that Polymorphism is the ability of an object to take on many forms. The most common use of polymorphism in OOP occurs when a parent class reference is used to refer to a child class object. Any Java object that can pass more than one IS-A test is considered to be polymorphic. A reference variable can be of only one type. Once declared, the type of a reference variable cannot be changed. There are two types of polymorphism in Java compile-time polymorphism and runtime polymorphism.

Runtime polymorphism: Runtime polymorphism a process in which a call to an overridden method is resolved at runtime rather than compile-time.

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

Ans:

```
package com.company;

public class Polymorphysim {
    public static void main(String[] args) {
        Animal animal;
        animal= new Dog();
        animal.eat();
        animal=new Cat();
        animal.eat();
        animal=new Lion();
        animal.eat();
    }
}
```

```
package com.company;
public class Animal {
    void eat() {System.out.println("eating...");}
}
```

```
package com.company;
public class Cat extends Animal {
    void eat() {System.out.println("eating rat...");}
}
```

```
package com.company;
public class Lion extends Animal {
    void eat() {System.out.println("eating meat...");}
}
```

```
package com.company;
public class Dog extends Animal {
    void eat() {System.out.println("eating bread...");}
}
```

```
eating bread...  
eating rat...  
eating meat...  
  
Process finished with exit code 0
```

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

Ans: 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. Objects in an OOP language provide an abstraction that hides the internal implementation details.

b) write a program on abstraction in java.

```
package com.company;

public class Cat extends Animal {

    public void animalSound() {
        System.out.println("The Cat says: meow meow");
    }
}
```

```
package com.company;

public class Main {

    public static void main(String[] args) {
        Cat cat= new Cat();
        cat.animalSound();
        cat.sleep();
    }
}
```

```
package com.company;

public class Main {

    public static void main(String[] args) {
        Cat cat= new Cat();
        cat.animalSound();
        cat.sleep();
    }
}
```

