Day: M T W T F S

Date: / /

Name : ZAHID-Ur-Rehman

Student ID : 16603

Department : BS (CS)

Submatted to : Sir Ayub khan

Semester : 2nd

Subject : oops

Date : 29/6/2020

Day: M T W T F S

Date: / /

Question 1

(a)

Access Modifiers in Java.

(1) Private access modifer.

(2) Role of private constructor.

(3) Default modifer.

(4) Protected access modifier.

(5) Public access modifer.

The two type of modifiers in Java access modifer and non-access modifiers.

The access modifier in Java specifies the accessibility or scope of a field, method, constructing or class.

we can change the access level of fields, constructors, method and class by applying the access the modifiers on it.

Date: __/__/__

Day: M T W T F S S

(1) **private**: The access level of private modified is only within the class. It cannot be accessed from outside the class.

(2) **Defult**: The access level of a default modified 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 defaults.

simple example of private access modified.

(1) class A {

(2) Private int data = 40

(3) Private in data msg () system out Println (" Hello Java"); }

(4) }

(5)

Day: M T W T F S Date: /__/__

```
(6) public class simple {
(7) public static void main
        (string args []) {
(8)  A obj = new A();
(9) System out println(obj.data);
        compile Time Error
(10) obj.msg();// compile Time Error
(11) }
(12) }
```

(2) Default

If you don't use any modifier. It is treated as default by default. The default modifier accessible only package. I cannot be accessed from outside the package. It provides more accibility than the private But it is more restrictive the Protected and Public.

Day M T W T F S Date: / /

Example of default access modifier.

(1) // save by A.java

(2) package pack;

(3) class A {

(4) void msg() {System.out.println ("Hello");}

(5) }

(1) save by B.java

(2) package mypack;

(3) import back *;

(4) class B {

(5) public static void main (String args[]){

(6) A obj = new A(); // compile Time Error

(7) obj msg(); // compile Time Error

(8) }

(9) }

Day: ☐ M ☐ T ☐ W ☐ T ☐ F ☐ S

Date: / / /

~~Question 1 (b)~~

Question 1 (b)

(1) class A {

(2) private int data = 400

(3) private in data msg()

system.out.println

("Hello Java"); }

(4) 3 4

(5)

6 Public class simple {

(7) public static void

main(String args[]) {

(8) A obj = new A();

(9) System.out.println
(obj.data);

compile Time Error

(10) obj.msg(); //
compile Time
Error

(11) }

(12) }

question 2 (a)
public and protect access modifier.

(3) Protected: The access level of a protected modifier is within the package and outside the package through child class.
If you do not make child the class
It cannot be accessed from outside package.

(4) public: The accessed level of pubilc a modifier is evey where.
It can be accessed from within class, out side class within the

There are many non access modifier, such as static, abstract, sychronized, native, volatile, transient etc.
Here we are going to learn the access modifier only.

Example of protected access modifier.

(1) // save by A.java
(2) package pack;
(3) public class A {
(4) protecd void msg() {System.out.println("Hello");}
(5) }

(1) // save B.java
(2) package my pack;
(3) import Pack*;
(4)
(5) class B extends A {
(6) pubic static void main

Day: M T W T F S Date: / / /

```
(string args [] E
(7) B obj = new B ();
(8) obj msg ();
(9) }
(10) }
(4) public.
```

Example of public access modifier.

```
(1) // save by A.java
(2).
(3) package class A {
(4) public class A {
(5) public void msg ()
{System.out println("Hello");}}
```

(1) // save by B. java

(2).

(3) pack my pack.

(4) import pack.*;

(5).

(6) class B {

(7) public static void main

args []) {

(8) A obj = new A();

Day: M T W T F S

(11)

Date: /__/__

Question 2

(b)

(1) // save by A java

(2) Package pack;

(3) class A {

(4) void msg() { System.
out println
("Hello"); }

(5) }

(1) Save by B java

(2) Package my pack;

(3) import back *

Day: M T W T F S S Date: / /

```
(4) class B {
(5) public static
        void main
    (String args[]) {
(6) B obj = new
    A(); //
        compile Time Error
(7) obj msg(); //
        compile Time
        Error
(8) }
(9) }
```

Question No 3

(a)

Inhertance: The inhertance can be defined as the process whose one class acquires (methods and fields) of another.
with use of inhertance the information and messageable in a heraial order.
The class which the perperties of other is known as subclass and the class whose properted are inhertance is know subclals

Extendy keywords.

The entends is the keyword used to inherit the properties of a class.

Simple code:

~~Following code~~

The simple code following is an example a demonstraning Java inhetence. In this example class namely calculations my calculation.

using extends keyword the my calculation inherits the methods addition () and subraction () of calculation class. copy and paste the following in a file with name, my calucation, java,

Question 3 (b)

inheritance class Animal

class mammal extends

Animal {

}

class Reptile extends Animal {

}

public class Dog extends

mammal {

}

public static void main(

String args[]) {

mammal a = new Animal();

mammal m = new Animal

();

Dog d = new Dog();

system.out.println(m instanceof

Animal);

Day: M T W T F S Date: __/__/__

```
system.out.println(d instanceof
                    mammal);
System.out.println(d instanceof
                    Animal);
    }
}
```

This will product following result

output

Question 4

(a)

what is Polymorphism?

Polymorphism is the ability of an object of to take on a many forms.

The most common use of a polymorphism in oop occurs when a parent class reference is used to a child class object. Any Java object that can pass more than one IS-A test is considered to be the polymophic In Java.

all Java object are Polymorphic.

since any object will
to access Pass the
IS-A test for their
own type and for
the class object.
It is important to know
that the only possible
way to access an object
through a reference variable.
The reference variable
can be reasigned to
other object provides
that it is not declared
final.

A reference variable can
refers to any object
of its declared type.
Its declared type a reference
variable can be declared
as as class
Example.

Day: M T W T F S (19) Date: / /

Question 4 (b)

```
/* File name : Employee.java */
public class Employee {
   private String name;
   private String address;
   Private int number;
   public Employee (String name
   adress, int number) {
      this. name = name;
      this address = address;
      this. number = number

   public void mail check () {

      System.out.println ("miling a
      check to " + this.name + " " +
      this. address);

   public string to string () {
         return name;
```

```
(String new address) {

address = new address

}

public get number () {

return number;

}
```

Now suppose we extend Employee class as follows.

```
/* File name : salary.java */

public class salary extends

Employee {

Private double salary;
```

```
// Annual Salary

public Salary (String name,
String address, int number,
double salary) {

public void mailCheck() {
System.out.println("within
mailcheck of Salary
class ");
System.out.println("mailing
check to " + get
Name()
+ " with salary " + salary);
}
public double getSalary() {
return salary;
```

```
public void setSalary(
    double newSalary) {

    if (newSalary >= 0.0) {

    }

}

public double computePay() {

System.out.println("computing
 extra salary pay for " +
        getName());

    return salary/52;

}
```

Question 4 b

Now, you study the following program carefully and try to determine its output.

```
/* File name : VirtualDem.java */

public class virticalDemo {

public static void main
(String [] args) {

Salary s= new salary("mond
Mutasim", Ambeta, up",
3, 3600.00);

Employee s=3 new salary
("John Adams", "Boston, MA", 2,
2400.00);
```

```
system.out.println("call mailcheck
        using salary reference--");
    s.mailchecks();

system.out.println("In call
    mailcheck using Employee
    reference--");
    e.mailcheck();
}

}
```

This will product the following result

output

contracting an Employee

contracting an Employee

Question 5 (a)

Why is abstraction ? in used oop,

As per dictionary, abstraction is the equality of a dealing with ideas rather then event.

For example, when you consider the case of e-mail, complex details such as what happens as soon as you send an e-mail the protocol your e-mail sever uses for used are hidden from the user. The send an e-mail you just need to type to the content, mention, the address of the receiver.

Day: M T W T F S Date: __/__/__ Day:

mention the address of the reciever and click send.

Abstract class

A class which contains the abstract keyword in it declaration is known as abstract class.

Abract class may or may not contain abstract know.

But if a class has at least one abstract method. then the clall must be declared abstract.

To use an abstract class. you have to inhait it form another clall provides implementation to the abstrut method.

Day: M T W T F S Date: / /

Question 5 (b)

```
// Java program to
illustrate the
// concept of Abstraction

abstract class Shape
{
    String color;
// these are abstract methods
~~public~~ abstract double area();
Public abstract string to
    string ();
//abstract class can have
    constructor
Public Shape (string color){
```

Day: M T W T F S Date: __/__/__

```
System.out.println("shape
        contructor called");
    this.color = color;
}

//
    this is concrete
        method
public string getColor() {
    return color;
}

}

class circle extends
        shape

    double radius;
public circle(String color,
        double radius) {
```

Day: M T W T F S Date: / / /

```
Public circle (String color,
                double, radius) {
//
    calling Shape constructor
    super (color);
    System (color);
    System.out.println ("circle
        constructor called");
    double area ()
    return math.PI * math.
        pow(radius, 2);
    }
    override
    Public String to string() {
        return circle color is
```

```
" + super.color +
        " and area is : " + areaC();
    }
}
class
    Rectangle extends shape{

double length;
double width;

Public Rectangle(String color,
        double color.
    output:
shape constructor called
circle constructor called
shape constructor called
Rectangle constructor called

Rectangle is Reded area is 15.20
Rectangle color is yellowed
```