**OOPS & Java Fundamentals**

Total Questions: 25

Emp Id: **1012745**

Name: **Amay Mandloi**

**Q1) A method defined in a superclass is redefined in a subclass with an identical method signature is called\_\_\_\_\_\_\_\_\_\_\_.**

A. Method overloading

B .Late binding

**C. Method overriding**

D. Dynamic binding

**Q2) Consider the code below and choose the correct option.**

class GameShape {

public void displayShape() {

System.out.println("displaying shape");

}

// more code

}

class PlayerPiece extends GameShape {

public void movePiece() {

System.out.println("moving game piece");

}

// more code

}

public class TestShapes {

public static void main (String[] args) {

PlayerPiece shape = new PlayerPiece();

shape.displayShape();

shape.movePiece();

}

}

A. GameShape class inherits the generic displayShape() method

**B. PlayingPiece class inherits the generic displayShape() method**

C. GameShape class inherits the generic movePiece() method

D. PlayingPiece class inherits the generic movePiece() method

**Q3) A class can inherit instance variables and methods from a more abstract superclass.**

**A .True**

B. False

**Q4) Constructor can have Return Type**

**a) True**

**b) False**

**Q5) The inheriting class cannot override the definition of existing methods by providing its own implementation.**

A. True

**B. False**

**Q6) Given below the sample code:**

class Hotel {

public int bookings=2;

public void book() {

bookings++;

}

}

public class SuperHotel extends Hotel {

public void book() {

bookings--;

}

public void book(int size) {

book();

super.book();

bookings += size;

}

public static void main(String args[]) {

SuperHotel Shotel = new SuperHotel();

Shotel.book(2);

System.out.print(Shotel.bookings);

}

}

**Find the output of the following code :**

**A. Compile error**

B. No Output

C.2

D.4

**Q7) HAS-A relationships are based on inheritance, rather than usage.**

A. True

**B. False**

**Q8) At run-time, a Java program is nothing more than objects 'talking' to \_\_\_\_\_\_\_\_\_\_\_.**

A. Other binders

B. Other objects

**C. Other classes**

D. Other methods

**Q9) Consider the below code and choose the correct output.**

public class Main {

public int a;

public long b;

public void test(long b)

{

System.out.println("long b");

}

public void test(int a)

{

System.out.println("int a");

}

public static void main(String[] args) {

Main e=new Main();

e.test(9\*1000000000);

}

}

A. Error

**B. Int a**

C. Long b

D. Long a

**Q10) If you do not have access to the source code for a class, but you want to change the way a method of that class works, then could you use subclassing to do that that is to extend the "bad" class and override the method with your own better code?.**

**A. True**

B. False

**Q11) Subclassing polymorphism is sometimes called "true polymorphism".**

**A .True**

B. False

**Q12) If you do not have access to the source code for a class, but you want to change the way a method of that class works, then could you use subclassing to do that that is to extend the "bad" class and override the method with your own better code?**

**A.True**

B.False

**Q13)The methods in class object are(choose four)**

A.Concat

B.Equals

C.Notify

D.Compare

E.Clone

F.Wait

**Q14) An interface cannot have an inner class.**

A.True

**B.False**

**Q15) Examples of class are( choose 3)**

A. White

B. Person

C. Classroom

D. Length

E. Car

**Q16) Given below the sample code :**

1 class Hotel {

2 public int bookings;

3 public void book() {

4 bookings++;

5 }

6 }

7 public class SuperHotel extends Hotel {

8 public void book() {

9 bookings--;

10 }

11 public void book(int size) {

12 book();

13 super.book();

14 bookings += size;

15 }

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

17 Hotel hotel = new Hotel();

18 hotel.book(2);

19 System.out.print(hotel.bookings);

20 }}

How can we correct the above code ? (choose all that apply)

A. By removing argument '2' at line number 18.

B. **By creating object of "SuperHotel" subclass at line 17 & calling book(2) from it at line 18**

C. No correction needed.

D. By adding argument "int size" to the method book at line number 3.

**Q 17) Method overloading is done during \_\_\_\_\_\_\_.**

A. **Program compilation**

B. Dynamic binding

C. Runtime

**D**. Late binding

**Q18) If no access modifier is specified in Java the default access modifier assumed is**

a) private

b) public

c) protected

**d) package level**

**Q20) Which of these is the valid declarations for the main method?**

A. public void main();

B. **public static void main (String args[])**

C. public static void main(String);

D. public static int main(String args[])

**Q21) A function can be abstract and final at the same time.**

A. True

**B. False**

**Q22) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ is the concept by which one class is inherited from more than one super class**

A. **Multiple inheritance**

B. Mutilevel inheritance

C. Single inheritance

D. None of the above

**Q23) A class can inherit instance variables and methods from a more abstract superclass.**

**A. True**

B. False

**Q24) To make salary in the following class definition read-only:**

class Employee

{

double salary;

}

a good approach would be to

a) Make the employee class private

b) Make salary protected

**c) Make salary private and define a method called getSalary()**

d) Make salary private and define methods called getSalary() and setSalary()

**Q25) Polymorphism is one interface with \_\_\_\_\_\_\_\_\_\_.**

A. Multiple record

B. Single method

C. **Multiple methods**

D. Single record