

```
1. Public class Demo{  
    Void main(){  
        System.out.println("JAVA");  
    }  
    Static void main(String args){  
        System.out.println("Spring");  
    }  
    Public static void main(String[] args){  
        System.out.println("Hibemate");  
    }  
    Void main(Object[] args){  
        System.out.println("Apache Camel");  
    }  
}
```

What is the output?

- 1. Hibernate
 - 2. Spring
 - 3. JAVA
 - 4. Apache Camel
-

```
2. Class Employee{  
    Public final void show(){  
        System.out.println("show() inside Employee");  
    }  
}
```

```
Final class unit extends Employee {  
    Public void show1() {  
        Final int x=100;  
        System.out.println("show() inside Unit");  
    }  
}
```

```

        System.out.println(x);
    }
}

Public class Demo11 {
    Public static void main(String[] args) {
        Employee employee = new Unit();
        New Unit.show1();
    }
}

```

3. What will be the output when the above code is complied and executed?

- A. 100
 - i. Show() inside Unit
- B. Show() inside Employee
- C. Show() inside Unit
 - 1. Show() inside Unit
 - ii. 100
- D. Show() inside Unit**
 - i. 100

4. What is the result of the following code?

```

Public class Bank {
    Static Class Customer {
        Public void go() {
            System.out.println("Inside Customer");
        }
    }
}

```

```

}

Static Class Account extends Customer {

    Public void go() {

        System.out.println("Inside Account");

    }

}

Static Class Branch extends Customer {

    @Override public void go() {

        System.out.println("Inside Branch");

    }

}

Public static void main(String[] args) {

//Line 1

}

}

```

5. What will be the output when we add the below code at Line1 and execute the program?

```

Customer customer = new Account();

Branch branch = (Branch) customer;

branch.go();

```

1. Inside Customer
2. Inside Account
3. Inside Branch
4. The Code does not compile because (Branch)Customer is incorrect
5. An exception is thrown at runtime because (Branch)Customer is incorrect

6. Predict the output of the following code.

```
Public class Game {  
    Public static void main(String[] agrs) {  
        displayRegistration("Hockey"); //Line 1  
        displayRegistration("Kho-Kho", 132, 102, 36); //Line 2  
    }  
}
```

```
Public static void displayRegistration (String gameName, int.. id) {  
    System.out.println("Registration for "+ gameName + ".");  
    for(int i=0; i<id.length; i++) {  
        System.out.println(id[i] + " ");  
    }  
}  
}
```

1. Registration for Hockey:

Hockey

Registration for Kho-Kho:

Kho-Kho

132 102 36

2. Registration for Hockey'

Registration for Kho-Kho:

132 102 36

3. Registration for Hockey:

4. Registration for Hockey:

Hockey

5. Public interface InterfaceDemo {

```
//Line 1  
}
```

Select the suitable code fragment can be inserted at Line1. (Choose all that apply.)

(Checkbox)

1. Void display (int x);
2. Void display (int x){
 }
3. Public static void display(int x){
 }
4. default void display(int x){
 }
5. public interface Demo {
 }

6. What is the output of the following code?

```
Class Employee {
```

```
    Void disp(char c){  
        System.out.print("Employee name starts with : "+c+".");  
        System.out.print("His experience is : 11 years. ");  
    }  
}
```

```
Class Main extends Employee {
```

```
    Void disp(Char c) {  
        Super.disp(c);  
        System.out.print("Another employee name also starts with : "+c+".");  
        new Employee().disp("D");  
        disp(7);  
    }
```

```

String disp (int c) {
    System.out.print("His experience is :" +c+ );
    return "Bye";
}
}

Public class Demo11 {
    Public static void main (String a[]){
        Employee emp = new Main();
        emp.disp("S");
    }
}

```

1. Employee name starts with : S. His experience is : 11 years. Another employee name also starts with : S. Employee name starts with : D. His experience is : 11 years. His experience is : 7.
 2. Employee name starts with : S. His experience is : 11 years. Another employee name also starts with : S. His experience is 7 years
 3. Employee name starts with : S. His experience is : 11 years. Another employee name also starts with : S. Employee name starts with : D. His experience is
 4. Employee name starts with : S. His experience is : 11 years. Another employee name also starts with : S.
-

7. Predict the output of the below code:

```
Class Dog{
```

```
    Void show(){
```

```
        System.out.print("Dog");
```

```
}
```

```
}
```

```
Class Cat{
```

```
    Void show(){
```

```
        System.out.print("Cat");
```

```
}
```

```

}

Class BullDog extends Dog{

    Void show(){

System.out.print("BullDog");

}

}

Public class Test {

    Public static void main (String[] args) {

System.out.print("Implementing type Casting");

Dog d = new Dog();

BullDog bd = (BullDog) d;

bd.show();

}

}

```

1. Display "Implementing type Casting" in Console.
 2. Display "Implementing type Casting" and "BullDog" in Console.
 3. RUNTIME ERROR : java.lang. ClassCastException
 4. Display "BullDog" in console.
-

8. Given:

```

Public class ExcepDemo{

    Public static void main(String [] args)

    {

        try

        {

            method();

            System.out.print("Inside try");

        }

        Catch (RuntimeException ex)

```

```

{
    System.out.print("Inside catch(RuntimeException)");
}
Catch (Exception ex1)
{
    System.out.print("Inside catch(Exception)");
}
finally
{
    System.out.print("finally");
}
System.out.print("end");
}

Public static void method()
{
    //Line 26
}
}

```

Which code fragment can be inserted at Line 26 to display the output as “Inside catch(RuntimeException) finally end” ?

- 1. throw new RuntimeException();**
 2. throw new Exception();
 3. throws new RuntimeException();
 4. throws new Exception();
 5. throw new Error();
-

9. Given:

```

Public class ExceptionInClass
{

```

```
int data=10;

void calculate() throws Exception
{
    try
    {
        data++;
    try
    {
        data++;
        // Line12
    }
    Catch(Exception ex)
    {
        data++;
    }
    Catch(Exception ex)
    {
        data++;
    }

}
Void display()
{
    System.out.println(data);
}

Public static void main(String[] args)
{
    ExceptionInClass exceptionInClass = new ExceptionInClass();
```

```

        exceptionInClass.calculate();

        exceptionInClass.display();

    }

}

```

Which of the below code fragment needs to be inserted at Line12 to display the output as 15.

1. try{
 data++;
 throw new Exception();
 }Catch(Exception ex){
 data++;
 throw new Exception();
 }

 2. try{
 data++;
 throw new Exception();
 }Catch(Exception ex){
 }

 3. try{
 throw new RunTimeException();
 }Catch(Exception ex){
 data++;
 throw new RunTimeException();
 }

 4. try{
 throw new Exception();
 }Catch(Exception ex){
 data--;
 throw new Exception();
 }

 5. None of the above
-

10. What is the output when the below code is compiled and executed?

```

import java.util.regex.Matcher;
import java.util.regex.Pattern;

```

```
public class Demo1{
    public static void main(String[] args){
        Pattern pattern=Pattern.compile("x*y");
        Matcher match=pattern.matcher("y");
        Boolean boolean1=match.matches();
        System.out.println(boolean1);
    }
}
```

- a. True
- b. Compilation error
- c. False
- d. Y

11. Given the below code snippet, predict the correct option

```
Public class Operator{
    Public static void main(String[] args){
        float val1=5.3f;
        float val2=2.3f;
        float result= val1 %val2;
        System.out.println(result);
    }
}
```

- a. Code compiles, runs and produces the output 0.7000003
- b. Compilation fails because % operator cannot be applied on float data range
- c. An exception is thrown at runtime
- d. Code compiles, runs but no output

12. What is the result when the following code is completed and executed?

```
Class Light{
    Boolean isOn;
}

Void turnOn(){
    isOn=true;
}

Void turnoff(){
```

```

    isOn=false;
}

}

Class LightDemo{
    Public static void main(String[]args){
        Light light1=new Light();
        Light light2=new Light();
        light1.turnOn();
        System.out.println(light1.isOn);
        light1.turnOff();
        System.out.println(light1.isOn);
        System.out.println(light2.isOn);}}
```

a. True

False

False (third item should be null and not false)

b. True

False

True

c. False

False

False

d. False

False

True

Ans : True

False

Null

13. What will be the output of the code given below?

```
Public class ABC{
```

```
    Public static void main(String[]args){
```

```

Boolean flag=false;

If (flag = true){

    System.out.println("true");}

If (flag = false){

    System.out.println("false");}}

```

- a. True
- b. False
- c. Compilation fails
- d. An exception is thrown at runtime

Ans : a. True

14. What is the result when the following code snippet is compiled?

```

Class Employee{
    Int employeeId;
    Double getEmployeeId(){
        System.out.println("Employee Id");
        Return employeeId;
    }
}

```

- A. The code will not be compiled as there is no main method
- B. The code will not be compiled as the return type in the getEmployeeId method should be int not double**
- C. The code will be compiled successfully and Employee.java file will be generated
- D. The code will be compiled successfully and Employee.class file will be generated

Ans : b. The code will not be compiled as the return type in the getEmployeeId method should be int not double

15. Public class Test{

```

    Public void method(){ //if this line is public static void method (), answer will be 012
        For(int i=0;i<3;i++){
            System.out.print(i);
        }
    }
}

```

```

Public static void main(String[]args){
    Method();
    Print(i);}}

```

- A. 012
- B. 0 1 2
- C. Compilation fails
- D. An exception is thrown at runtime

Ans : c. Compilation fails

16. What is the output of the below code snippet?

```
enum Customer{  
    private CUSTID;  
    public CUSTNAME;  
    protected ADDRESS;  
}
```

- a. An exception is thrown at runtime
- b. EnumNotDefined Exception
- c. No error
- d. Compilation fails

Ans : d. Compilation fails

17. What will be the output of the following code?

```
Public class Test{  
  
    Public void method(){  
  
        For(int i=0;i<3;i++){  
  
            System.out.print(i);  
  
            System.out.print(i);  
  
        }  
  
    }  
  
}
```

- a. 0123
- b. 012
- c. Compilation fails
- d. An exception is thrown at runtime

Ans : c. Compilation fails

18. What will be the output of the below code?

```
Public class Main{  
    Static int[] x;  
    Static{  
        X[0]=102;}  
    Public static void main(String[]args){  
        System.out.println(x); }  
}
```

- A. No output
- B. Compilation fails
- C. java.lang.ExceptionInitializeError
- D. java.lang.ArrayIndexOutOfBoundsException

Ans : c. java.lang.ExceptionInitializeError

19. Predict the output of the following code:

```
Class VarArgsDemo{  
    Static void func(int...x)  
    {  
        System.out.println("Number of arguments "+x.length);  
  
        For(int i:x)  
            System.out.print(i+" ");  
        System.out.println();  
    }  
    Void func(int a) //Line1  
    { System.out.println("one"); }  
  
    Public static void main(String[]args){  
        new VarArgdDemo().func(150);  
        func(11, 12, 13, 14);
```

```
func(); }  
}  
  
A. Number of arguments: 1  
150  
Number of arguments: 4  
11 12 13 14  
Number of arguments: 0
```

- B. One
Number of arguments: 4
11 12 13 14
Number of arguments: 0
- C. Number of arguments: 4
11 12 13 14
Number of arguments: 0
- D. Number of arguments: 4
11 12 13 14

Ans : b. One

```
Number of arguments: 4  
11 12 13 14  
Number of arguments: 0
```

20. Analyze the below code and predict the output

```
Class Employee{  
    Double salary;  
    Public static void main(String[]args){  
        Employee employee1=null;  
        Employee employee2=null;  
        employee1= new Employee();  
        employee2= new Employee();  
        employee1= new Employee();  
        Employee employee3=null;  
        employee1= employee3=null; //Line8  
        System.out.println("Hello World");  
    } }
```

How many objects are eligible for garbage collection after executing line8?

- A. 3
- B. 0
- C. 2
- D. 1

Ans : a. 3

21. Which statements are true about the following code snippet?(choose all that apply)

```
Public class Developer{}  
Public class Employee{  
    Public String empName;  
}  
  
Public class Tester extends Employee{  
    Public Developer developer;  
}  
  
Public class Testing extends Tester{}
```

- A. Testing has a empName
- B. Testing has a Developer
- C. Testing is a Developer
- D. Testing is a Employee
- E. Tester is a Testing
- F. Employee has a Developer

Ans : d & e & a

22. Observe the below code snippet:

```
Public class BasePlant{  
    Private String name;  
    Public BasePlant(String name){  
        this.name=name;}  
}  
Public String getName(){  
    return name;  
}  
}  
Public class Tree extends BasePlant{  
    Public void growFruit(){}
```

```
Public void dropLeaves(){}
}
```

Which of the following statement is true?(choose all that apply)

- A. The code will be compiled without any changes
- B. The code will be compiled only if the below code is added to the Tree class

```
Public Tree() {super("Plant");}
```
- C. The code will be compiled only if the below code is added to the BasePlant class

```
Public BasePlant() {Tree(); }
```
- D. The code will be compiled only if the below code is added to the BasePlant class

```
Public BasePlant() {this("Plant"); }
```

Ans : b & d

23. Predict the output of the following code:

Apple.java

```
Public class Apple{
    Public void color(){
        System.out.println("Red");
    }
}
```

Mango.java

```
Class Mango extends Apple{
    @Override
    Public void color(){
        System.out.println("Yellow");
    }
    Public static void main(String[]args){
        Apple apple=new Mango(); //Line1
        apple.color(); //Line2
    }
}
```

- a. Red Yellow
- b. Yellow Red
- c. Compilation fails because of an error in Line 2
- d. Yellow

Ans : d.Yellow

24. Given:

```
Public interface interfaceDemo{  
    //Line1  
}
```

Select the suitable code fragment can be inserted at Line1(choose all that apply)

- a. void display(int x);
- b. void display(int x){}
- c. public static void display(int x){}
- d. default void display(int x);
- e. public interface Demo{}

Ans : a & c

25. Analyze the below code and select the suitable outcome

```
Class Apple{  
    Private Apple() //Line1  
    {  
        System.out.println("Apple constructor")  
    }  
    void display(){  
        System.out.println("Apple constructor")  
    }  
}  
Public class Main{  
  
    Public static void main(String[]args){  
  
        Apple apple=new Apple(); //Line2  
        apple.display();  
    } }
```

- a. Apple constructor
Hello World
- b. Runtime exception is thrown at Line1
- c. Runtime exception is thrown at Line2

d. Unresolved compilation problem: The constructor Apple() is not visible

Ans : d. Unresolved compilation problem: The constructor Apple() is not visible

26. Predict the output of the below code:

```
Public class Demo{  
    Static int x=232;  
    Int y=135;  
  
    Public void display(){  
        System.out.print("Inside Demo")  
    }  
  
    Public static void staticMethod(){  
        System.out.print(x); //Line 8  
    }  
  
    Public static void main(String[]args){  
        Demo staticMethod(); //Line 13  
  
        Demo demo=new Demo();  
  
        demo.staticMethod(); //Line15  
  
        staticMethod();  
  
        demo.display(); //Line 16  
    } }
```

- A. 232 135 Inside Demo 232 135 Inside Demo232 135 Inside Demo
- B. 232232232Inside Demo
- C. 232232Inside Demo
- D. 232232

Ans : b. 232232232Inside Demo

27. What is the output when the below code is compiled and executed?

```
Class ExDemo{  
    Public static void main(String[] args){
```

```

Try{
    Throw 110;}
}
Catch(int ex){
    System.out.println("Caught Exception" + ex);
}
a. Caught the Exception as 10
b. Caught the Exception as 0
c. Compilation fails
d. An exception is thrown at runtime

```

Ans : c. Compilation fails

28. What will be the output of the below code?

```

Class Student
{
    String stuName="Jacklin";
    void display(){
        try{
            stuName+="John";
            func();
        }
        Catch(Exception e){
            stuName+="GoodName";
        }
    }

    void func() throws Exception{
        try{
        }
        Catch(Exception e){
            stuName+="GoodName";
        }
    }

    void func() throws Exception{
        try{
            stuName+=" ";
            method();
        }
        Catch(Exception e){
            Throw new Exception();
        }
    }
}

```

```

finally{
    stuName+="!!!";
}

stuName+="hello"
}

void method() throws Exception{
    throw new Exception();
}

void disp(){
    System.out.println("stuName");
}
}

Public static void main(String [] args){
    try{
        Student student=new Student();
        student.display();
        student.disp();
    }
    Catch(Exception e){
        System.out.println("Catch block");
    }
}
}

```

- a. JacklinJohn !!!hello
- b. JacklinJohn!!!hello
- c. JacklinJohn!!!helloGoodName Catch block
- d. JacklinJohn !!!GoodName**

Ans : d. JacklinJohn !!!GoodName

29. Predict the output for the below code?

```

Public class TestDemo{
    Public static void main(String[] args){
        Int sum, a=10, b=10;
        Try{
            System.out.println(sum=a/b);
            Return; //Line 1
        } catch(ArithmaticException | Exception e){ //Line2
    }
}

```

```

        System.out.println(e.getMessage());
    }finally{
        System.out.println("in finally");
    }
}
}
}

```

- a. Compilation fails because of the error in Line1
- b. Prints:
/by zero
In finally
- c. **Compilation fails because of the error in Line2**
- d. Program compiles successfully but not prints anything in console

Ans : c. Compilation fails because of the error in Line2

30. Given:

```

Public class ExcepDemo{
    Public static void main(String[] args){
        Try{
            Method();
            System.out.print("Inside Try");
        }
        Catch (RuntimeException ex){
            System.out.print("Inside catch(Runtime Exception)");
        }
        Catch (Exception ex1){
            System.out.print("Inside catch(Exception)");
        }
        Finally{
            System.out.print("finally");
        }
        System.out.print("end");
    }

    Public static void method(){
        //Line 26
    }
}

```

Which code fragment can be inserted at line26 to display the output as “inside catch(RuntimeException)
finally end”?

- a. throw new RuntimeException();
- b. throw new Exception();
- c. throws new RuntimeException();
- d. throws new Exception();
- e. throw new Error();

Ans : a. throw new RuntimeException();

31. What is the output of the following code ?

Package exceptions;

```
Import java.io.*;
Public class ExceptionDemo{
    Static class Car implements AutoCloseable{
        Public void close(){
            System.out.print("Automatic Door Close");
        }
    }
    Static class carWindow implements Closerable{
        Public void close(){
            System.out.print("CarWindow");
            throw new RuntimeException();
        }
    }

    Public static void main(String[] args){
        Try(Car car=new Car()){
            CarWindow carWindow=new CarWindow(){
                System.out.print("Inside try block");
            }
        Catch(Exception e){
            System.out.print("Inside catch block");
        }
        Finally{
            System.out.print("finally");
        }
    }
}
```

- A. Automatic Door close CarWindow Inside try block inside catch blockfinally
- B. Automatic Door Close CarWindow Inside catch blockfinally
- C. Inside try blockCarWindowAutomatic Door CloseInside catch blockfinally
- D. An exception is thrown at run time
- E. Compilation fails

Ans : c. Inside try blockCarWindowAutomatic Door CloseInside catch blockfinally

32. What is the output of the following code?

```
class Person{  
    public Person(String name){  
        System.out.println(name);  
    }  
}  
  
public class Student extends Person{  
    public Student(){      //Line 8  
        System.out.println("Inside Student");  
    }  
  
    public static void main(String[] args) { // Line 11  
        new Person("Jacklin");  
    }  
}
```

- 1) JacklinInside Student
- 2) Jacklin
- 3) Compilation fails because of an error in Line 8
- 4) Compilation fails because of an error in Line 11

Ans : Compilation fails because of an error in Line 8

33. Given:

```
public abstract class Employee {  
    private int empld;  
    private int salary;  
    public abstract void display();  
    public void setValues(int empld, int salary){  
        this.empld = empld;  
        this.salary = salary;  
    }  
}
```

Which of the following classes provide the right representation of the child class of Employee class?

- 1) public abstract class Child extends Employee {
 private int z;
}
- 2) public class Child implements Employee {
 private int z;
}
- 3) public class Child extends Employee {
 private int z;
 public void display();
}
- 4) public class Child extends Employee {
 private int z;
 public void display() {
 /* code here */
 } }

Ans : 4) public class Child extends Employee {
 private int z;
 public void display() {
 /* code here */
 } }

34. Given an abstract Class Customer as below:

```
public abstract class Customer
{
    public abstract String getCustomerType();
}
```

Select a Valid implementation of getCustomer Type method in another class, from the below options:

- 1) abstract class C1 extends Customer{
 public string getCustomer Type()
 { return "Premium";
 }
}
- 2) Customer customer = new Customer(){
 public String getCustomerType()
 { return "Premium";
 }
}
- 3) class C1 extends Customer{
 public String getCustomerType()
 { return "Premium";
 }
}
- 4) new Customer(){
 public String getCustomerType()
 { return "Premium";
 }
}

Ans : 3) class C1 extends Customer{

```
public String getCustomerType()
{ return "Premium";
}
```

35. Analyze the below code and predict the output

```
class Employee{  
  
    double salary;  
  
    public static void main(String args[]){  
        Employee employee1=null;  
        Employee employee2=null;  
        employee1 = new Employee();  
        employee2 = new Employee();  
        employee2 = employee1;  
        employee1 = new Employee();  
        Employee employee3=employee1;  
        employee1=employee3=null; //Line 8  
        System.out.println("Hello world");  
    }  
}
```

How many objects are eligible for garbage collection after executing Line 8?

- 1) 3
- 2) 0
- 3) 2
- 4) 1

Ans : 1) 3

36. class Customer{

```
    int customerId = 11201;  
    Customer() {  
        customerId = 11240;  
    }  
}
```

```
class Main {  
    public static void main(String args[]){  
        Customer customer = new Customer();  
        System.out.println(customer.customerId);  
    }  
}
```

- 1) 11201
- 2) 11240
- 3) **Compilation fails**
- 4) An exception is thrown at run time

Ans 3)Compilation fails

37. Select the suitable code to be inserted in line 1 and line 2 to get the below output Line 2 should be used to change the space into tab space

False

Simple

Demo

For

Regular

Expressions

Using

Pattern

Matching

Simple demo for the regular expression using pattern matching

```
Public class RegExDemo1{  
  
    Public static final String string1="Sample demo for"+ "regular expressions" + "usingpatternmatching"  
  
    Public static void main(String[] args){  
  
        //Line 1  
  
        //Line2  
    }
```

}

- a)

```
System.out.println(String1.matches("\t"));
String[] splitString=(String1.split(\s+));
for(String string splitString){
    System.out.println(String);
}
System.out.println(string1.replaceAll("\S+","\t"));
```
- b)

```
System.out.println(String1.matches("\t"));
String[] splitString=(String1.split(\s));
for(String string splitString){
    System.out.println(String);
}
System.out.println(string1.replaceAll("\S+","\t"));
```
- c)

```
System.out.println(String1.matchesAll("\t"));
String[] splitString=(String1.split(\s));
for(String string splitString){
    System.out.println(String);
}
System.out.println(string1.replaceAll("\S+","\t"));
```
- d)

```
System.out.println(String1.matchesAll("\t"));
String splitString=(String1.split(\s));
for(String string splitString){
    System.out.println(String);
}
System.out.println(string1.replaceAll("\S+","\t"));
```

38. What is the result of attempting to complete and run this program?

```
Class Demo1{
Public static void main(String[] args){
    String c="a",//Line 3
    Switch(c)//Line4
    Case 65 //Line5
        System.out.println("One");
    Break
        Case"a"/Line6
    System.out.println("two");
    Case 3//line 10
        System.out.println("three");
```

```

Break
}
}}
a) two
b) two three
c) Runtime exception is thrown at Line5
d) Computation fails because of an error in Line 5 and Line 10
e) Error in Line 4 .cant accept string to static

```

39. Select all possible options that are valid among the following Enums can be defined inside_____

- a) An interface
- b) A class {Multiple choice question}
- c) A static Context
- d) A method

40. Given the Enum definition and the Java class

```

Enum Day{
    SUNDAY(1),MONDAY(2),TUESDAY(3),WEDNESDAY(4),THURSDAY(5),FRIDAY(6),SATURDAY(7)
    Private int value
    Private Day(int value){
        This.value=value;
    }
    Public int getvalue(){
        Return this.value
    }
}
Public class TestEnum{
    Public static void main(String[] args){
        For(Day day: Day values()){
//Line1
    }
}

```

what should be placed at line 1 to get the output as shown below:Choose as that apply

SUNDAY-MONDAY-TUESDAY-WEDNESDAY-THURSDAY-FRIDAY-SATURDAY

- a) System.out.print(day.toString()+"-")
- b) System.out.print(day.getvalue()+"-")
- c) System.out.print(day.name()+"-")
- d) System.out.print(day.getName()+"-")

41. What is the result when the following code is completed and executed?

```

Public class Test{
    Public void method(){
        for(int i=0;i<3;i++){

```

```

        System.out.print(i)
    }
}

Public static void main(String[] args){
    Method();
}

a) 012
b) 0 1 2
c) compilation fails
d) An exception is thrown at runtime

```

42. What will be the result when the below code is completed and executed?

```

Import java.util.regex.Pattern;

Public class RegExDemo2{
    Private static final string String1=“
    Private static final string String2=“one two three four five”;
    Public static void main (String[] args){
        Pattern pattern=Pattern compile(String1)//Line 7
        String[] strArr=pattern split(String2)//Line 8
        For(String str:strArr){
            System.out.println(str);
        }
    }

A)Compilation fails because of an error in Line8
B)Compilation fails because of an error in Line7
C)one
two
three
four
five
D)An exception is thrown at run time

```

43. What is the output of the code given below

```

Public class ABC{
    Public static void main(String args[]){
        Boolean flag=false;
        If(flag=true){
            System.out.println(“true”)
        }
    }
}

```

```

}
If(Flag==false){
System.out.println("false");
}
}
a) true
b) false
c) Compilation fails
d) An exception is thrown at runtime

```

44. Given

```

Class Demo2{
    Public static void main(string[] args){
        Int[]X={111,112,113,114,115,116,117,118,119,110}
        //Line1
        System.out.println("count is"+i);
    }
}
}

```

Which is the correct code fragment to be inserted at Line 1 to execute the code to print count starts from 111,112,113....

- a)for(int i=0;i<=x;length i++){
- b)for(int i:x){**
- c)for(int x: i){
- d)for(int i: x.length){

45. what is the output when the following code is compiled and executed?

```

Class Calculator
{
    Int a=123;
    Int b=200;
    Public void display(){
        Sysytem.out.println("a"+a+"b"+b+"")
    }
}
Class CalculatorDemo
{
    Public static void main(String[] args)
    {
        Calculator calculator1=new Calculator();//Line1
        Calculator calculator2= Calculator1//Line2
    }
}

```

```

Calculator1.a+=1;
Calculator1.b+=1;

System.out.println("calculator1 values")
Calculator1.display()
System.out.println("calculator2 values")
Calculator2.display()
}
a)calculator1 values
a.124 b.201
b) calculator2 values
a.125 b.202
c) calculator1 values
a.124 b.201
d) calculator2 values
a.123 b.200
e) calculator1 values
a.124 b.201
f) calculator2 values
a.124 b.201
g)compilation fails because of the error in Line 2

```

48. what is the output when the following code is compiled and executed?

Abstract class Customer

```

{
Public int custId
Customer()
{
custId=23456;
}
abstract public void setId()
abstract final public void getId()
}
```

Class Demo extends Customer

```

{
Public void setId(int custId)
{
This.custId=custId
}

Final public void getId()
{
System.out.println("Customerid"+custId)
}

Public static void main(String[] args)
{
Demo demo=new Demo()

Demo.setId(1102)

Demo.getId()
}

}

a) compilation fails because of an error in Line9
b) compilation fails because of an error in Line11
c) Runtime exception is thrown at line 17
d) CustomerId 1102
e) compilation fails because of an error in Line17

```

49.What is the output for the following code?

```

Class Person{
Public Person(String name){
System.out.println(name)
}
}

Public class Student extends Person{
Public Student(){ //Line8

```

```

        System.out.println("Inside Student");
    }
    Public static void main(String[] args){ //line11
        New Person("Jacklin");
    }
}

a) JacklinInside Student
b) Jacklin
c) Compilation fails because of the error in line 8
d) Compilation fails because of the error in line 11

```

50. Given

```

Public class ConstructorDemo(){

Private int id;

Private final String name;

Static final int age=22;

ConstructorDemo1(int I,Stringn){

Id=1;

name=n;

}

ConstructorDemo1(inti,String n,int a){

Id=1;

name=n;

}

Void display(){

System.out.println(id+” +name+” +age);

}

Public static void main(String args[]){

//Line1

//Line2

ConstructorDemo1.display();

```

```
ConstructorDemo2.display();
}
}
```

The below code fragment can be inserted at Line 1 and Line 2.What will be the output?

```
ConstructorDemo1 constructorDemo1=new ConstructorDemo1(1101,"Jacklin");
ConstructorDemo1 constructorDemo2=new ConstructorDemo2(1102,"John",25);
```

- a) 1101 Jacklin 22
 1102 John 25
- b) 1101 jacklin 22
 1102 John 22
- c) An exception throws at the runtime due to initialization of the variable constructor at Line 2
- d) Compilation fails because of the error in line2 that variable cant be executed

51. Given

```
Class Employee{
    Public final void show(){
        System.out.println("show()inside Employee");
    }
}

Final class Unit extends Employee{
    Public void show1(){ //Line1
        Final int x=100
        System.out.println("show()inside Unit");
        System.out.println(x);
    }
}

Public class Demo11{
    Public state void main(string[] args){
        Employee employee=new Unit();
```

```
New Unit().show1();  
}  
}
```

What will be the output when the above code is computed and executed?

- a) 100
 show()inside Unit
- b) Show()inside Employee
- c) Show()inside Employee
 Show()inside Unit
 100
- d) Show()inside Unit
 100

52.Given

```
Class Parent{  
}  
Class Child extends Parent{  
}  
Final class GrandChild extends Child{  
}
```

Which of the following statement is not true about the above code?

- a) The above code represents the multi-level inheritance with the two level
- b) The GrandChild class can Access the protected and public members of the parent and child class
- c) Instance of parent class can accept the reference of the child class but not the reference of
 GrandChild class
- d) The GrandChild class can override the methods of both Parent class and Child class

53.Predict the output of the below code

```
Public class InnerClassDemo  
{  
    InnerClassDemo()  
    {  
        System.out.print("InnerClassDemo Constructor")  
    }  
    Class Demo  
    {  
        Demo()  
        {  
            System.out.println("Demo Constructor");  
        }  
    }  
}
```

```

Public void disp()
{
System.out.print("Simple Class");
}
}
Public static void main(String[] args)
{
InnerClassDemo innerClassDemo=new InnerClassDemo();
innerClassDemo.createDemo();
}
Void createDemo()
{
(new Demo() {}).disp();
}
}

```

- a) Compilation fails
- b) An exception is thrown at the runtime
- c) Prints InnerClassDemo ConstructorDemo ConstructorSimple class
- d) Print Demo ConstructorSimple class

54. Analyze the code below and select the suitable outcome

```

Class Apple{
Private Apple() //Line1
{
System.out.println(:Apple Constructor");
}
Void display(){
System.out.println("Hello World");
}
}
Public class Main{
Public static void main(string[] args){
Apple apple=new Apple() //Line 2
Apple.display();
}
}

```

- a) Apple Constructor
Hello World
- b) Runtime exception is thrown at the line1
- c) Runtime exception is thrown at the line2
- d) Unresolved computation problem .The constructorApple() is not visible

55. what will be the output for the below code

```
Class Parent
{
Void message()
{
System.out.println("Inside parent class");
}
}

Class Derived extends Parent
{
Void message()
{
System.out.println("Inside derived class");
}
Void display()
{
message();
super.message();      //Line1
}
}
```

```
Class SuperDemo
{
Public static void main(String args[])
{
Derived derived=new Derived();
Derived.display();      //Line2
}
}
```

- a) Inside parent class
 Inside derived class
 - b) Inside derived class
 - c) Inside parent class
 - d) Inside derived class
 Inside parent class
-

56. which of the below exceptions are mostly thrown by JVM in a Java application?(Choose all that apply)

- a) ClassCastException
- b) IllegalStateException
- c) NumberFormatException
- d) IllegalArgumentException

e) ExcdeptionInitializerError

57. What is the output when the below code is computed and executed/

```
Class ExDemo{  
    Public static void main(String[] args  
    )  
    {  
        Try{  
            Throw 110;  
        }  
        Catch(int ex){  
            System.out.println("Caught Exception"+ex);  
        }  
    }  
}
```

- a) Caught the Exception as 10
- b) Caught the Exception as 0
- c) Compilation fails
- d) An exception is thrown at runtime

58. Given:

```
Public class ExceptionClass  
{  
    Int data=10;  
    Void calculate()throws Exception  
    {  
        Try  
        {  
            Data++;  
        Try  
        {  
            Data++  
            //Line12  
        }  
        Catch(Exception ex)  
        {  
            Data++;  
        }  
    }catch(Exception ex){  
        Data++;  
    }  
    Void display()  
    {
```

```

System.out.println(data);
}
Public static void main(String[] args) throws Exception
{
ExceptionClass exceptionClass=new ExceptionClass;
exceptionClass.calculate();
exceptionClass.display();
}
}

```

Which of the below fragment needs to be inserted at the Line12 to display the output as 15

- a) try{
 data++;
 throw new Exception();
 }
 Catch(Exception ex){
 data++;
 throw new Exception();
 }
- b) try{
 data++;
 throw new Exception();
 }
 Catch(Exception ex){
 }
- c) try{
 throw new Exception();
 }
 Catch(Exception ex){
 data++;
 throw new Exception();
 }
- d) try{
 throw new Exception();
 }
 Catch(Exception ex){
 data--;
 throw new Exception();
 }
- e) Correct: try{
 data++;
 throw new Exception();
 }

```
Catch(Exception ex){  
    data++;  
    throw new Exception();  
}finally{  
    Data++; }
```

59. Given:

```
Publilc class ExcepDemo  
{  
Public static void main(String[] args)  
{  
try  
{  
methods();  
System.out.print("Inside try");  
}  
Catch(RuntimeException ex)  
{  
System.out.print("Inside catch (Runtime Exception)")  
}  
Catch(Exception ex1)  
{  
System.out.print("Inside catch(Exception)");  
}  
Finally  
{  
System.out.print("finally");  
}  
System.out.print("end");  
}  
Public static void method()  
{  
//Line 26  
}  
}
```

Which code fragment can be inserted at Line 26 to display the output as "Inside catch(RuntimeException)finally end"?

- a) Throw new RuntimeException();
- b) Throw new Exception();
- c) Throws new RuntimeException();
- d) Throws new Exception();
- e) Throw new Error()

60. Given:

```
Public class ExceptionDemo1{  
    Static class Car implements AutoCloseable{  
        Public void close(){  
            System.out.print("Car door close");  
            Throw new RuntimeException();  
        }  
    }  
    Static class CarWindow implements Closeable{  
        Public void close(){  
            System.out.println("Car window close");  
            Throw new RuntimeException();  
        }  
    }  
    Public static void main(String[] args){  
        Try{  
            //Line 1  
        }  
        Catch(Exception e){  
            System.out.println("Catch exception");  
        }  
        Finally{  
            System.out.print("finally");  
        }  
    }  
}
```

Which of the below code can be inserted at Line1 to display THE OUTPUT AS “try block finally” (Choose all that apply)

A) `Car car=new Car();`
`CarWindow carWindow=new CarWindow();`
`System.out.print("try block");`

B) `Car car=new Car();`
`System.out.print("try block");`

C) `Car car=new CarWindow()`
`System.out.print("try block");`

D) `system.out.print("try block")`

61. Which two statements are true for a two-dimensional array?

- A.It is implemented as an array of the specified element type
- B.Using a row by column convention, each row of a two-dimensional array must be of same size
- C.At declaration time, the number of elements of the array in each dimension must be specified
- D.All the methods of the class Object may be invoked on the two-dimensional array

- a) Option (A) and (B)
- b) Option (A) and (D) [per Gova A & D are the true statements]**
- c) Option (B) and(C)
- d) Option (C) and (D)

62. Observe the following code:

```
Public class WrapperClassDemo{  
    Public static void main(String aa[]){  
        {  
            Int x=90;  
            Integer i1=new Integer(x);  
            Int y=90;  
            Integer i2=new Integer(y);  
            System.out.print(i1.compareTo(i2)+" "+Integer.compare(i2,i1)+" "+i1.equals(i2)+" "+(i1==i2));  
        }  
    }  
}
```

In the above code snippet identify which of the following method compares the given values and return an int which tells lesser or greater

- a) Compare()**
- b) Equals()
- c) compareTo()**
- d) ==

63. Predict the output of the below code

```
Public class TestDemo{  
    Public static void main(String[] args){  
        String value1="Hello"  
        String value2=new String("Hello");  
        System.out.println(value1.equals(value2)+" ,"+(value1==value2))  
        String value3=value2.intern()  
        System.out.println((value1==value3)+" ,"+value1.equals(value3));  
    }  
}
```

}

A)false true

 True true

B)true false

 True false

C)true true

 True false

D)true true

 False true

Output: true false true true

64. Predict the output for the below code

```
Public class TestDemo{  
    Public static void main(String[] args){  
        Integer n1=new Integer(100);  
        Integer n2=new Integer(100);  
        Integer n3=127;  
        Integer n4=127;  
        Integer n5=128;  
        Integer n6=128;  
        Int n7=129;  
        Int n8=129;  
        System.out.print(n1==n2);  
        System.out.print(n3==n4);  
        System.out.print(n5==n6);  
        System.out.print(n7==n8);  
    }  
}
```

a) **False true false true**

b) False true true true

c) False true false false

d) False false false true

65. What is the result of attempting to compute and run this code snippet?

```
TreeSet treeset=new TreeSet();  
treeset.add("first");  
treeset.add("First");  
treeset.add("Second");  
system.out.println(treeset.ceiling("Fir"))
```

- a) Fir
- b) first
- c) First
- d) St
- e) Compilation fails
- f) An exception throws at the runtime

66. Which of the following statements are true if a duplicate element obj T is added to a HashSet?

- a) The element obj T is not added and add() method returns false
- b) The element obj T is added successfully
- c) An exception occurs during runtime
- d) An exception occurs during compile time

67. Predict the output for the below code snippet?

```
Pulic class TestDemo{  
Public static void main(String[] args){  
ArrayList Strings=new ArrayList()  
Strings add("aAaA");  
Strings add("AaA");  
Strings add("aAa");  
Strings add("AAaa");  
Collections sort(Strings);  
For(String string:Strings){  
System.out.print(string);  
}  
}  
}
```

- a) Compilation fails
- b) aAaA aAa AAaa Aaa
- c) AAaa AaA aaa aAaA
- d) AaA AAaa aAaA aAa
- e) An exception is thrown at runtime

68. Given:

```
public class Group extends TreeSet{  
public static void main(String[] args){  
Group g=new Group();
```

```

g.add(new Person("Hans"));
g.add(new Person("Jane"));
g.add(new Person("Hans"));
System.out.println("Total"+g.size());
}
public boolean add(Object o){
System.out.println("Adding"+o);
return super.add(o);
}
}
class Person{
private final String name;

public Person(String name){
this.name=name;
}

public String toString(){
return name;
}
]

```

what will be the output when this code snippet is compiled and executed?

- a) Adding Hans
An exception is thrown at the runtime
- b) Adding Hans
Total 3
- c) Adding Hans
Total 2
- d) The code does not compile**

69. What is true regarding the following code snippet?

```

interface StaticInterface
{
static void staticMethod()
{
System.out.println("Inside interface");
}
}
class StaticInterfaceImpl implements StaticInterface
{

```

```

public void staticMethod()
{
    system.out.println("Inside class");
}
}

public class StaticDemo
{
    public static void main(string[] args)
    {
        new StaticInterfaceImpl().staticMethod();
    }
}

```

- a) Code will not be compiled as the static method always be public
- b) Code will not be compiled as the static method is overridden in staticInterfaceImpl
- c) code will print "inside interface" on execution
- d) code will print "inside class" on execution**

70. What will be the execution result of the following code?

```

public class Formatting
{
    public class void main(String[] args)
    {
        LocalDate date=LocalDate.of(2016,11,13);
        DateTimeFormatter formatter= DateTimeFormatter.ofPattern("dd/MMM/YYYY",Local UK);
        system.out.println(date.format(formatter));
    }
}

```

- a) execution will not be successful as the month is not in a valid range
- b) compilation will not be successful as the month is not in a valid range
- c) 13/NOV/2016 will be printed
- d) 13/Nov/2016 will be printed**

71. which of the following is incorrect regarding interfaces in Java SE8

- a.all the methods are public,abstract by default
 - b.all the variables are public by default
 - c.methods can have implementation
 - d.its possible to hold static method
-
- a) a and b
 - b) b and c**
 - c) a,b and c

d) a only

72. Refer the below code snippet and predict the output.

```
interface Interface1
{
    default void method1()
    {
        system.out.println("Inside default method");
    }
}
interface DefaultExtends extends interface1
{
    default void method1()
    {
        system.out.println("Default method redefined");
    }
}
public class InterfaceWithDefaultMethod implements DefaultExtend
{
    public static void main(String[] args)
    {
        InterfaceWithDefaultMethod defaultExtend=new InterfaceWithDefaultMethod(); //Line4
        defaultExtend.method1(); //Line5
    }
}
```

- a) Inside default method
- b) Default method redefined**
- c) Compilation fails at Line5
- d) Runtime exception will be thrown at Line5

73. what happens if “default “ keyword is omitted while defining a default method in interface?

```
interface Interface1
{
    void method1()
    [
        system.out.println("Inside default method");
    ]
}
a.method cannot be overridden in the implementing classes
b.method can be overridden in the implementing classes
c.method cannot be given body in the interface
d.compilation error occurs
```

- a) a and b

- b) a,b and c
- c) c and d
- d) b and c

74. Select the valid code fragment according to java coding standard?

- 1)

```
public void draw(String s){  
    if(s.equals("Square")){  
        drawSquare();  
    }  
    if(s.equals("Rectangle")){  
        drawRectangle();  
    }  
}
```
 - 2)

```
public void draw(String s){  
    if("Square".equals(s)){  
        drawSquare()  
    }  
    if("Rectangle".equals(s)){  
        drawRectangle();  
    }  
}
```
- only option(1) is valid
only option(2) is valid
Both(1) and (2) are valid
Both(1) and (2) are invalid

75. Which of the below are NOT good practices for creating objects?

- a) Lazy initialization of objects
- b) Creating String literals instead of String objects
- c) Creating Wrapper objects instead of primitives [googled and verified]
- d) invoking static factory methods for immutable classes

76. Identify the issue in the below code fragment

```
public class Ex1{  
    public String formatininput(String i){  
        if(i.trim().length()==9){  
            StringBuilder s1=new StringBuilder();  
            s1=s1.insert(0,"+1");  
            s1=s1.insert(6,"");  
            s1=s1.insert(10,"-");  
            return s1.toString();  
        }  
    }  
}
```

```

}
return null;
}
public static void main(String args[]){
Ex1 ob=new Ex1();
String l;
ob.formatInput(i);
}}

```

- a) compilation fails at Line3
- b) Compilation fails at Line 6
- c) Null pointer exception will be thrown if the value of l is null
- d) Compilation fails due to error in Line7

77. Which of the below statement indicate the need to use the factory pattern?

- a) we have two classes that do the same thing
- b) we only want one instance of the object to exist
- c) we want to build a chain of objects
- d) we don't want the caller to depend on a specific implementations

78. Consider the below code snippet

```

Locate locate=new Locate("da","DK");
NumberFormat numberFormat=NumberFormat.getInstance(locate);
String number=numberFormat.format(100,99);
System.out.println(number);

```

Here NumberFormat.getInstance() follows which design pattern?

- a) Factory method pattern
- b) Singleton pattern
- c) Abstract Factory pattern
- d) Builder pattern

79. Given:

```

//Assume all the required imports are added
public class TestDemo{
static int a=0;
static ArrayList b;
@BeforeClass
public static void beforeClass(){
a=10;
b=new ArrayList();
}
@Before

```

```
public void int(){
a=15;
b.add(a);
}
@Test
public void test(){
a=a+20;
system.out.print(a);
system.out.println(b);
}
@Test
public void test1(){
a=a+30;
system.out.print(a);
system.out.print(b);
}
}
```

Predict the output?

- a) 35[15]
45[15,15]
- b) 35[15]
65[15,15]
- c) 35[15]
45[15]
- d) 35[15]
65[15]
- e) 35[15]
65[30]

80.Predict the output for the below code

```
import static org.junit.Assert.assertEquals
import static org.junit.Assert.assertSame
import org.junit.Test

public class TestDemo{
@Test
public void teststringSame(){
String str="Junit";
assertSame("JunitTesting",str.concat("Testing"))
}
@Test
public void testingEqual(){}
```

```

String str="Junit";
assertEquals("JunitTesting",str.concat("Testing"));
}
}
public class TestDemo{
@Test
public void testingSame(){
String str="Junit";
assertSame("JunitTesting",str.concat("Testing"));
}
@Test
public void testingEqual(){
String str="Junit";
assertEquals("JunitTesting",str.concat("Testing"));
}
}

```

- a) Both the Testcases fail**
- b) Both the testcases run successfully
 - c) TestCase testingSame will be a success and testingEquals will be failure
 - d) TestCase testingSame will be a failure and testingEquals will be success
-

81. What is the output when the below code is compiled and executed?

```

import java.util.regex.Matcher;
import java.util.regex.Pattern;

public class Demo1{
    public static void main(String[] args){
        Pattern pattern=Pattern.compile("x*y");
        Matcher match=pattern.matcher("y");
        Boolean boolean1=match.matches();
        System.out.println(boolean1);
    }
}

```

- e. True**
- f. Compilation error

- g. False
- h. Y

82. What will be the output of the following code?

```
Public class Test{  
    Public static void main(String[] args){  
        Int [][] x;  
        x=new int[3][4];  
        for(int i=0;i<3;i+=2){  
            for(int j=0;j<4;j++){  
                x[i][j]=i+j;  
                System.out.print(x[i][j]+” ”);  
            }  
        }  
    }  
}
```

- a. 0 1 2 3 1 2 3 4 2 3 4 5
- b. 0 1 2 3 2 3 4 5**
- c. 0 2 1 3 2 4
- d. 0 2 2 4

83. Given the below code snippet, predict the correct option

```
Public class Operator{  
    Public static void main(String[] args){  
        float val1=5.3f;  
        float val2=2.3f;  
        float result= val1 %val2;  
        System.out.println(result);  
    }  
}
```

- A. Code compiles, runs and produces the output 0.7000003**
- B. Compilation fails because % operator cannot be applied on float data range
- C. An exception is thrown at runtime
- D. Code compiles, runs but no output

84. What is the result when the following code is completed and executed?

```
Class Light{  
    Boolean isOn;  
}
```

```

Void turnOn(){
    isOn=true;
}

Void turnoff(){
    isOn=false;
}

}

Class LightDemo{
    Public static void main(String[]args){
        Light light1=new Light();
        Light light2=new Light();
        light1.turnOn();
        System.out.println("light1 is on");
        light1.turnOff();
        System.out.println(light1.isOn);
        System.out.println(light2.isOn);}}
```

e. True
False
False

f. True
False
True

g. False
False
False

h. False
False
True

True
False
Null

85. What will be the output of the code given below?

```
Public class ABC{  
    Public static void main(String[]args){  
        Boolean flag=false;  
        If (flag = true){  
            System.out.println("true");}  
        If (flag = false){  
            System.out.println("false");}}}
```

- e. True
- f. False
- g. Compilation fails
- h. An exception is thrown at runtime

86. What is the result when the following code snippet is compiled?

```
Class Employee{  
    Int employeId;  
    Double getEmployeeId(){  
        System.out.println("Employee Id");  
        Return employeId;  
    } }
```

- A. The code will not be compiled as there is no main method
- B. The code will not be compiled as the return type in the getEmployeeId method should be int not double
- C. The code will be compiled successfully and Employee.java file will be generated
- D. The code will be compiled successfully and Employee.class file will be generated

87. Public class Test{

```
    Public void method(){  
        For(int i=0;i<3;i++){  
            System.out.print(i);  
        }  
    }  
  
    Public static void main(String[]args){  
        Method();}}
```

- A. 012
- B. 012
- C. Compilation fails
- D. An exception is thrown at runtime

88. What is the output of the below code snippet?

```
enum Customer{  
    private CUSTID;  
    public CUSTNAME;  
    protected ADDRESS;  
}
```

- A. An exception is thrown at runtime
- B. EnumNotDefined Exception
- C. No error
- D. Compilation fails

89. What will be the output of the following code?

```
Public class Test{  
  
    Public void method(){  
  
        For(int i=0;i<3;i++){  
  
            System.out.print(i);  
  
            System.out.print(i);  
        }  
    }  
}
```

- A. 0123
- B. 012
- C. Compilation fails
- D. An exception is thrown at runtime

90. What will be the output of the below code?

```

Public class Main{
    Static int[] x;
    Static{
        X[0]=102;}
    Public static void main(String[]args){
        System.out.println(x); }
}

```

- A. No output
- B. Compilation fails
- C. java.lang.ExceptionInitializeError
- D. java.lang.ArrayIndexOutOfBoundsException

91. Predict the output of the following code:

```

Class VarArgsDemo{
    Static void func(int...x)
    {
        System.out.println("Number of arguments "+x.length);

        For(int i:x)
            System.out.print(i+" ");
        System.out.println();
    }
    Void func(int a) //Line1
    { System.out.println("one"); }

    Public static void main(String[]args){
        new VarArgdDemo().func(150);
        func(11, 12, 13, 14);
        func(); }
}

```

- A. Number of arguemnts: 1
150
Number of arguemnts: 4

11 12 13 14
Number of arguments: 0

B. One

Number of arguments: 4
11 12 13 14
Number of arguments: 0

C. Number of arguments: 4

11 12 13 14
Number of arguments: 0

D. Number of arguments: 4

11 12 13 14

92. Analyze the below code and predict the output

```
Class Employee{  
    Double salary;  
  
    Public static void main(String[]args){  
  
        Employee employee1=null;  
        Employee employee2=null;  
        employee1= new Employee();  
        employee2= new Employee();  
        employee1= new Employee();  
        Employee employee3=null;  
        employee1= employee3=null; //Line8  
        System.out.println("Hello World");  
    } }
```

How many objects are eligible for garbage collection after executing line8?

- A. 3
- B. 0
- C. 2
- D. 1

93. Which statements are true about the following code snippet?(choose all that apply)

```
Public class Developer{}  
Public class Employee{  
    Public String empName;  
}
```

```
Public class Tester extends Employee{  
    Public Developer developer;  
}
```

```
Public class Testing extends Tester{}
```

- A. Testing has a empName
- B. Testing has a Developer
- C. Testing is a Developer
- D. Testing is a Employee
- E. Tester is a Testing
- F. Employee has a Developer

94. Observe the below code snippet:

```
Public class BasePlant{  
    Private String name;  
    Public BasePlant(String name){  
        this.name=name;  
    }  
    Public String getName(){  
        return name;  
    }  
}  
Public class Tree extends BasePlant{  
    Public void growFruit(){}
    Public void dropLeaves(){}
}
```

Which of the following statement is true?(choose all that apply)

- A. The code will be compiled without any changes
- B. The code will be compiled only if the below code is added to the Tree class

```
Public Tree() {super("Plant");}
```
- C. The code will be compiled only if the below code is added to the BasePlant class

```
Public BasePlant() {Tree();}
```
- D. The code will be compiled only if the below code is added to the BasePlant class

```
Public BasePlant() {this("Plant");}
```

95. Predict the output of the following code:

Apple.java

```
Public class Apple{
    Public void color(){
        System.out.println("Red");
    }
}
```

Mango.java

```
Class Mango extends Apple{
    @Override
    Public void color(){
        System.out.println("Yellow");

    }
    Public static void main(String[]args){
        Apple apple=new Mango(); //Line1
        apple.color(); //Line2
    }
}
```

- e. Red Yellow
- f. Yellow Red
- g. Compilation fails because of an error in Line 2
- h. Yellow**

96. Given:

```
Public interface interfaceDemo{
    //Line1
}
```

Select the suitable code fragment can be inserted at Line1(choose all that apply)

- A. void display(int x);**
- B. void display(int x){}
- C. public static void display(int x){}**
- D. default void display(int x);
- E. public interface Demo{}

97. Analyze the below code and select the suitable outcome

```
Class Apple{
```

```

Private Apple() //Line1
{
System.out.println("Apple constructor")
}
void display(){
System.out.println("Apple constructor")
}
}
Public class Main{

Public static void main(String[]args){

    Apple apple=new Apple(); //Line2
    apple.display();
} }

```

- A. Apple constructor
Hello World
- B. Runtime exception is thrown at Line1
- C. Runtime exception is thrown at Line2
- D. Unresolved compilation problem: The constructor Apple() is not visible**

98. Predict the output of the below code:

```

Public class Demo{
    Static int x=232;
    Int y=135;

    Public void display(){
        System.out.print("Inside Demo")
    }

    Public static void staticMethod(){
        System.out.print(x); //Line 8
    }

    Public static void main(String[]args){
        Demo staticMethod(); //Line 13
        Demo demo=new Demo();
        demo.staticMethod(); //Line15
    }
}

```

```
staticMethod();  
demo.display(); //Line 16  
} }
```

- A. 232 135 Inside Demo 232 135 Inside Demo232 135 Inside Demo
- B. 232232232Inside Demo
- C. 232232Inside Demo
- D. 232232

99. What will be the output of the below code snippet?

```
Package com.infy;  
Class Pet{  
Public void displayName(){  
    System.out.println("Inside Pet")  
}  
}
```

```
Package java.pack1;  
Import com.infy.Pet;  
Public class Demo{  
Public static void main(String[] args){  
Pet pet=new Dog();  
pet.displayName();  
}  
}
```

- A. inside Pet
- B. Inside Dog [to be the answer if there is a public class named “pet” in the code]
- C. inside Pet Inside Dog
- D. Compilation fails [per gova, if inside dog has to be the answer, there should be a public class pet in code]

100. What is the output when the below code is compiled and executed?

```
Class ExDemo{  
Public static void main(String[] args){  
    Try{  
        Throw 110;  
    }  
    Catch(int ex){  
        System.out.println("Caught Exception" + ex);  
    } } }
```

- e. Caught the Exception as 10
- f. Caught the Exception as 0
- g. Compilation fails**
- h. An exception is thrown at runtime

101. What will be the output of the below code?

```
Class Student
{
    String stuName="Jacklin";
    void display(){
        try{
            stuName+="John";
            func();
        }
        Catch(Exception e){
            stuName+="GoodName";
        }
    }

    void func() throws Exception{
        try{
        }
        Catch(Exception e){
            stuName+="GoodName";
        }
    }

    void func() throws Exception{
        try{
            stuName+=" ";
            method();
        }
        Catch(Exception e){
            Throw new Exception();
        }
        finally{
            stuName+="!!!";
        }
    }

    stuName+="hello"
}
```

```

void method() throws Exception{
    throw new Exception();
}

void disp(){
    System.out.println("stuName");
}
}

Public static void main(String [] args){
    try{
        Student student=new Student();
        student.display();
        student.disp();
    }
    Catch(Exception e){
        System.out.println("Catch block");
    }
}
}

e. JacklinJohn !!!hello
f. JacklinJohn!!!hello
g. JacklinJohn!!!helloGoodName Catch block
h. JacklinJohn !!!GoodName

```

102. Predict the output for the below code?

```

Public class TestDemo{
    Public static void main(String[] args){
        Int sum, a=10, b=10;
        Try{
            System.out.println(sum=a/b);
            Return; //Line 1
        } catch(ArithmaticException | Exception e){ //Line2
            System.out.println(e.getMessage());
        }finally{
            System.out.println("in finally");
        }
    }
}

```

- A. Compilation fails because of the error in Line1
- B. Prints:
/by zero
In finally
- C. Compilation fails because of the error in Line2**
- D. Program compiles successfully but not prints anything in console

103. Given:

```
Public class ExcepDemo{
    Public static void main(String[] args){
        Try{
            Method();
            System.out.print("Inside Try");
        }
        Catch (Runtimeexception ex){
            System.out.print("Inside catch(Runtime Exception)");
        }
        Catch (Exception ex1){
            System.out.print("Inside catch(Exception)");
        }
        Finally{
            System.out.print("finally");
        }
        System.out.print("end");
    }

    Public static void method(){
        //Line 26
    }
}
```

Which code fragment can be inserted at line26 to display the output as “inside catch(RuntimeException) finally end”?

- A. throw new Runtimeexception();**
- B. throw new Exception();
- C. throws new Runtimeexception();
- D. throws new Exception();
- E. throw new Error();

104. What is the output of the following code ?

Package exceptions;

```
Import java.io.*;
Public class ExceptionDemo{
    Static class Car implements AutoCloseable{
        Public void close(){
            System.out.print("Automatic Door Close");
        }
    }
    Static class carWindow implements Closerable{
        Public void close(){
            System.out.print("CarWindow");
            throw new RuntimeException();
        }
    }
}

Public static void main(String[] args){
    Try(Car car=new Car();
        CarWindow carWindow=new CarWindow()){
        System.out.print("Inside try block");
    }
    Catch(Exception e){
        System.out.print("Inside catch block");
    }
    Finally{
        System.out.print("finally");
    }
}
```

- A. Automatic Door close CarWindow Inside try block inside catch blockfinally
- B. Automatic Door Close CarWindow Inside catch blockfinally
- C. Inside try blockCarWindowAutomatic Door CloseInside catch blockfinally
- D. An exception is thrown at run time
- E. Compilation fails

105. Given:

```
Public class TestString3{
    Public static void main(String[] args){
        //insert code here//Line3
        System.out.println(s); }}
```

Which of the below code fragment when inserted independently at line3, generate the output as 498?

- a. String s="123456789"; s=(s-"123").replace(1,3,"24")-"89";
- b. StringBuffer s=new StringBuffer("123456789").s.delete(0,3).replace(1,3,"98").delete(3,8);
- c. StringBuffer s=new StringBuffer("123456789").s.substring(3,6).delete(1,3).insert(1,"24");
- d. StringBuffer s=new StringBuffer("123456789").s.substring(3,6).delete(1,2).insert(1,"24");

106. Identify which of the following class breaks its input into tokens using a whitespace pattern?

- a. InputStreamReader
- b. Console
- c. Scanner
- d. BufferedReader
- e. DataInputStream

107. What will be the output of the following code when it is compiled and executed?

```
Public class Hello{  
    Public static void main(String[] args){  
        String s="How\"are\"you?";  
        System.out.println(s);  
    }  
}
```

- a. The output will be
How "are" you?
- b. The output will be
How \"are\" you?
- c. Compilation fails
- d. An exception is thrown at runtime

108. What will be the output of the following code:

```
Public class WrapperClassDemo{  
    Public static void main(String aa[]){  
        Integer intWrapper=Integer.valueOf("12345");  
        Integer intWrapper2=Integer.valueOf("11",2);  
        Integer intWrapper3=Integer.valueOf("E",16);  
        System.out.println(intWrapper+" "+intWrapper2+" "+intWrapper3);  
    }  
}
```

- ```
 }
}
a. 12345 13 14
b. 12345 11 14
c. 12345 3 14
d. 12345 3 15
```

109. Given:

```
Public class Demo11{
 Public static void main(String args[]){
 1. Set numbers=new HashSet();
 2. numbers add(new Integer(45));
 3. numbers.add(88);
 4. numbers.add(new integer(77));
 5. numbers.add(null);
 6. numbers.add(789L);
 7. Iterator iterator=numbers iterator();
 8. while(iterator.hasNext())
 9. System.out.print(iterator.next());
 10. }}
```

Which of the following statements are true?

- a. Runtime exception will be thrown
- b. The output is 4588null789
- c. The output is 45887null789
- d. There is a compiler error on line1
- e. There is a compiler error on line7
- f. The output is null789884577

110. Public static void main(String[] args){  
 HashMap props=new HashMap<>();  
 props.put("key45","some value");  
 props.put("key12","some other value");  
 props.put("key39","yet another value");  
 Set s=props.keySet();  
 //Line1  
}

Which of the below code has to be inserted at Line1, to sort the keys in the props HashMap variable?

- a. Arrays.sort(s);
- b. S=new TreeSet(s);
- c. Collections.sort(s);

d. S=new SortedSet(s);

111. Predict the output for the below code snippet?

```
Public class TestDemo{
 Public static Collection get(){
 Collection sorted=new LinkedList();
 sorted.add("B");
 sorted.add("C");
 sorted.add("A");
 return sorted;
 }
 Public static void main(String[] args){
 for(Object obj: get()){
 System.out.print(obj+".");
 }
 }
}
```

- a. A, B, C
- b. B, C, A**
- c. Compilation fails
- d. The code runs with no output

112. What will be your observation on the below code snippet? (Assume that the class Item exists and choose all possible options)

```
Public class TestDemo{
 Public static void main(String[] args){
 TreeSet tset=new TreeSet();
 tset.add(new item());
 TreeSet b=tset; } }
```

- a. Compilation fails**

- b. An exception is thrown at runtime
- c. Compiles successfully with a warning
- d. Compiles and runs successfully without any warnings

113. Given:

```
Class Apple{
 A obj;
 Apple(A obj) {this.obj=obj; }

 public A getObject() {return this.obj; }
}
```

```
Class Main
{
 Public static void main(String[] args){
 //Line1
 }
}
```

Which of the following code snippet can be inserted at line1 to display the output as  
76  
Hello

- a. 

```
Apple apple=new Apple(76);
System.out.println(apple.getObject());
Apple appleObj=new Apple("Hello");
System.out.println(appleObj.getObject());
```
- b. 

```
Apple apple=new Apple(76);
System.out.println(apple.getObject());
Apple appleObj=new Apple("Hello");
System.out.println(appleObj.getObject());
```
- c. 

```
Apple apple = new Apple(76);
System.out.println(apple.getObject().toString());
Apple appleObj=new Apple("Hello");
System.out.println(appleObj.toString());
```

- d. Apple apple=new Apple(76);  
System.out.println(apple.getObject().toString());  
Apple appleObj;  
appleObj=apple;  
System.out.println(appleObj.toString());
114. Refer the below code snippets and predict the outcome?

```
Public class RepeatingAnnotations{
 @Retention(RetentionPolicy.RUNTIME)
 public @interface Chocolates{
 Favourite[] value() default();
 }

 @Favourite("Diary Milk")
 @Favourite("Kit Kat")
 @Favourite("5 star")
 @Favourite("Galaxy")
 public interface Chocolate{
 }

 @Repeatable(value=Chocolates.class)
 Public @interface Favourite{
 String value();
 }

 Public static void main(String[] args){
 Favourite[] a=Chocolate.class.getAnnotationsBy.Type(Favourite.class);
 Chocolates chocolates=Chocolate.class.getAnnotation(Chocolates.class); //Line5
 for(Favourite favourite: chocolates value()){
 System.out.println(favourite.value()); } } }
```

- a. Nothing will be displayed  
b. null will be printed  
c. Runtime exception will be thrown at Line 5  
d. Dairy Milk

Kit Kat  
5 Star  
Galaxy

115. What will happen to the following code when trying to get compiled?

```
Class RepeatableAnnotation{
 @SuppressWarnings("all") //line1
 @SuppressWarnings("deprecation") //line2
 public void over()
 {
 New Date().setDate(00); } }
```

- a. Unreachable code error will be generated at line2
- b. Compilation will not be successful as @SuppressWarnings annotation is non-repeatable in nature
- c. Warning will be issued as it is totally unnecessary to mention @SuppressWarnings("deprecation")
- d. Code will get compiles successfully without any warning

116. What will be the output when the following code is compiled and executed?

```
Public class TestDemo{
 Public static void main(String[] args){
 LocalDateTime date1=LocalDatetime.of(2017,Month.FEBRUARY, 11, 15, 30); //Line1
 LocalDateTime date2=LocalDateTime.of(2017, 2, 12, 10, 20);
 System.out.println(date1.compareTo(date2));
 } }
```

- a. -1 will be printed as execution result
- b. 1 will be printed as execution result
- c. Compilation error will be raised as the month is invalid in line1
- d. Exception will be raised as the month is invalid in line1
- e. None of the above

---

117. Given :

```
Public class TestString3{
```

```
Public static void main(String[] args){
 //insert code here//Line3
 System.out.println(s);
}
}
```

Which of the below code fragment when inserted independently at line 3 generate the output as 498?

- 1.String s="123456789",s=(s-"123")replace(1,3,"24")-"89";
- 2.StringBuffer s= new StringBuffer("123456789"),s.delete(0,3),replace(1,3,"98").delete(3,8);
- 3.StringBuffer s=new StringBuffer("123456789"),s.substring(3,6).delete(1,3).insert(1,"24")
4. StringBuffer s=new StringBuffer("123456789"),s.substring(3,6).delete(1,2).insert(1,"24")

118. Identify which of the following class breaks its input into tokens using a whitespace pattern?

- 1.InputStreamReader
- 2.Console
- 3.Scanner
- 4.BufferedReader
- 5.DataInputStream

119. Predict the output for the below code?

```
Public class TestDemo{
 Public static void main(String[] args){
 Int sum,a=10,b=10;
 Try{
 System.out.println(sum=a/b);
 Return; //line1
```

```
}catch (ArithmeticException | Exception e) //Line 2
System.out.println(e.getMessage());
}finally{
System.out.println("In finally");
}
}
```

1.compilation fails because of the error in Line 1

2.prints:

/by zero

In finally

3.compilation fails because of the error in Line 2

4.program compiles successfully but not prints anything in console

120. Given:

```
Public class Excepdemo
{
Public static void main(String[] args)
{
Try
{
Method();
System.out.print("inside try");
}
Catch(RuntimeException ex)
{
System.out.print("Inside catch(RunttimeException)");
}
```

```
}

Catch(Exception ex1){

System.out.print("Inside catch(Exception)");

}

Finally

{

System.out.print("finally")

}

System.out.print("end");

}

Public static void method()

{

//Line26

}

}
```

Which code fragment can be inserted at Line26 to display the output as "Inside catch(runtimeException)"

- 1.throw new RuntimeException();
- 2.throw new Exception();
- 3.throws new RuntimeException();
- 4.throws new Exception();
- 5.throw new Error();

121. What is the result of executing the following code?

```
Package exceptions;

Public class Demo

{

Public void division(int x,int y){
```

```

Try{
 Int z=x/y;
}

Catch(exception e){
 System.out.print("Arithmetic Exception")
}

Finally{
 System.out.print("finally block")
}

}

Public static void main(String[] args)
{
 Demo demo=new Demo();
 Demo division(0,8);
}
}

```

1.Arithmetic Exception Finally block  
**2.Finally block**  
3.Arithmetic Exception  
4.An exception is thrown at runtime

122. Given:

```

Public class ExceptionDemo1{
 Static class car implements AutoCloseable{
 Public void close()
 {
 System.out.print("Car door close")
 }
 }
}
```

```

 Throw new RuntimeException(); }

Static class CarWindow implements Closeable {
 Public void close()
 {
 System.out.print("Car door close")
 Throw new RuntimeException();
 }
}

```

```

Public static void main(String[] args){
 Try{
 //line1
 }
 Catch(Exception e){
 System.out.print("catch exception");
 }
 Finally{
 System.out.print("finally");
 }
}

```

Which one of below code can be inserted at Line1 to display the output as “try block finally”

1. Car car=new Car();
 CarWindow carWindow=new CarWindow();
 System.out.print("try block")
2. Car car=new Car();
 System.out.print("try block");
3. Car car=new CarWindow();
 System.out.print("try block");

4. System.out.print("try block"),

123. Void display()  
{  
    System.out.println("x= "+x+" y= "+y)  
}  
  
Public static void main(String[] args)  
{  
    ThisDemo thisDemo=new ThisDemo();  
    thisDemo.get().display()  
}  
}  
  
1.x=0 y=0  
2.x=45 y=56  
3.Compilation fails because of an error in Line 1  
4.Runtime exception is thrown at line 1

124. What is the output of below code?

```
Class MyException extends Throwable{
 Public MyException(String msg){
 Super(msg);
 }
}

Public class TestDemo{
 Static void myCode() throws MyException{
 Try{
 Throw new MyException("Test exception")
 }
 }
```

```

Catch(Error|Exception ex){
System.out.print("Inside Error and Exception")
} }

Public static void main(String[]args)throws MyException{
Try{
myCode();
}catch(Exception ex){
System.out.print("Inside Exception")
}
}
}

1.prints "Inside Error and Exception"
2.An Exception is thrown at runtime
3.Compilation fails
4.prints "Inside Exception"

```

125. Identify the output of the below code:

Class ThisDemo

```

{
Int x;
Int y;
ThisDemo(){
X=45;
Y=56;
}
ThisDemo get() //Line1
{
Return this;
}
```

```

Void display()
{
 System.out.println("x=*+x+y=*+y");
}

Public static void main(string[]args)
{
 ThisDemo thisDemo=new ThisDemo();
 thisDemo.get().display();
}
}

1.x=0 y=0
2.x=45 y=56
3.compilation fails because of an error at line 1
4.Runtime Exception is thrown at line 1

```

126. What will be the output of the following code?

```

Public class test{
 Public void method()
 {
 For(int i=0;i<3;i++){
 System.out.print();
 }
 System.out.print();
 }
}

```

1. 0123
2. 012
3. Compilation fails

4. An exception is thrown at runtime

127. What are the different types of memory areas used by JVM(choose two)?

1. Class

2. Heap

3. Stack

4. Queue

128. What is the output when the following snippet is compiled?

```
Class Apple{
 Int quantity;
}

Class Main{
 Public static void main (String args[]){
 Apple apple;
 System.out.println("apple quantity");
 }
}
```

1. 0 (Default value is printed)
2. The code will compile successfully and prints null
3. Compilation error variable might not have been initialized
4. Compilation error apple has not been initialized
5. apple quantity

129. What is the result of the following code is compiled and executed?

```
Class Calculator
{
```

```
Int a=123,b=200;

Public void display()
{
System.out.println("a: "+a+"b"+b+")
}

Class CalculatorDemo
{

Public static void main (String args[]){
 Calculator calculator1=new Calculator() //line1
 Calculator calculator2=calculator1; //line2
 calculator1.a+=1;
 calculator1.b+=1;
 System.out.println("calculator1.values");
 calculator1.display();
 System.out.println("calculator2.values");
 calculator1.display();
}
}

1. calculator1.values
a.124 b.201

calculator2.values
a.125 b.202

2. calculator1.values
a.124 b.201

calculator2.values
a.123 b.200

3. calculator1.values
a.124 b.201

calculator2.values
```

a.124 b.201

4. compilation fail because of an error in line2

130. JVM in java is a

- 1.Debugger
- 2Assembler
- 3.compiler
- 4.Interpreter

131. What is the result when the following code is executed?

```
Class Demo1{
 Public static void main (String args[]){
 Int i1=0;
 Int[] j={11,111,14,19,116,215}; //line4
 For (int i1:j) //line5
 System.out.print("%d",i1);
 }
}
```

- 1. 11  
111  
14  
19  
116  
215
- 2. 0  
1  
2

- 3
  - 4
  - 5
3. compilation fail because of an error in line5
4. Runtime exception is thrown at line 4

132. What is magic number in java in the context of java programming best practices?

- 1.A number which gets printed on the console
- 2.A direct usage of the number int the code
- 3.A number which magically disappears from the code
- 4.A number which is generated through error

133. Identify the issue in the below code:

```
Public class Student{
 Private School school;
 Private StudentDetails stuDetails;
 Private Fees fees;

 Public MarksHistory marksHistory(Marks marksDetails){
 //computation
 }
}
```

- 1.Issue: Single Responsibility principle (lazy initialization is the only other option but it's a best practice)
- 2.Issue: Character Encoding
- 3.Issue: Cycles between packages should be removed
- 4.Issue: Lazy Initialization

134. Predict the output of below code

```
Class Dog{
Void show(){
System.out.println("Dog");
}
}

Class Cat{
Void show{
System.out.println("Cat");
}
}

Class BullDog extends Dog{
Void show{
System.out.println("BullDog");
}
}

Public class Test{
Public static void main(String[]args){
System.out.println("Implementing type casting");
Dog d=new Dog();
BullDog bd=(BullDog)d;
Bd.show();
}
}
```

1. Displays " Implementing type casting" in console  
2. Displays " Implementing type casting" and "BullDog" in console  
3. RUNTIME ERROR: java lang ClassCastException  
4. Displays "BullDog" in console.

135. What is the output of the below code?

```
Public class Demo11{
 Public static void main(String[]args){
 Parent obj =new Child();
 Obj.display();
 }
}

Class Parent{
 Public void display(int a){
 System.out.println("Parent Method");
 }
}

Class Child extends Parent{
 Public void display()
 { System.out.println("Child Method");
 }
}
```

1.Compilation fails

2.Parent Method

3.Child method

4.An exception is thrown at runtime

136. Predict the output of the below code:

```
Class Employee{
 //....
```

```

}

Class Manager extends Employee{
 Public void someManagerMethod(){
 //...
 }
}

Class Officer extends Employee{
 //...
}

Public void someMethod(Employee e){
 Manager m=(Employee)e //Line 12
 m.someManagerMethod();
}

}

Class Demo{
 Public static void main(String s){
 Officer obj=new officer();
 Obj.someMethod(new Officer()); //Line 19
 }
}

```

- 1.Compilation fails because of an error in Line 12**
- 2.Compilation fails because of an error in Line 19
- 3.Runtime exception is thrown at line 12
4. Compilation fails because of an error in Line 12 and Line 19

137. Which statement is true about the classes and interfaces given below?

```

Public interface Demo1{
 Public void display(String points);
}

Public class Demo2 implements Demo1{

```

```
Public void display(String points){};
```

```
}
```

```
Public class Demo3{
```

```
 Public Demo1 disp(){
```

```
 //more code here
```

```
}
```

```
 Public string displayValue(){
```

```
 //more code here
```

```
}
```

```
}
```

```
Public class Demo4 extends Demo3{
```

```
 Public Demo2 disp(){
```

```
 //more code here
```

```
 Return null;
```

```
}
```

```
 Private String displayValue(){
```

```
 //more code here
```

```
}
```

```
}
```

1.compilation of class Demo2 will fail because of an error in line2

2.compilation of class Demo4 will fail because of an error in line2

3.compilation of class Demo4 will fail because of an error in line6

4.Compilation will succeed for all classes and interfaces

138. Predict the output of the below code:

```
Class VarArgsDemo
```

```
{
```

```
 Static void func(int ...x)
```

```
{
 System.out.println("Number of arguments "+x.length);
 For(int i:x)
 System.out.print(i+" ");
 System.out.println();
 }

Void func(int a) //line1
{
 System.out.println("one");
}

Public static void main(String[]args){
 New VarArgsDemo().func(150);
 Func(11,12,13,14);
 Func();
}
}
```

1.Number of arguments:1

150

Number of arguments:4

11 12 13 14

Number of arguments:0

## 2. One

Number of arguments:4

11 12 13 14

Number of arguments:0

## 3. Number of arguments:4

11 12 13 14

Number of arguments:0

4. Number of arguments:4

11 12 13 14

139. Given an abstract class customer below:

Public abstract class customer

{

Public abstract string getCustomertype();

}

Select a valid implementation of getCustomerType method in another class from below:

1. Abstract class C! extends Customer{  
    Public string getCustomerType()  
    {  
        Return"Premium";  
    }  
}
2. Customer customer=new Customer(){  
    Public string getCustomerType()  
    {  
        Return"Premium";  
    }  
}

140. What will be the output for the below code

Class Parent

{

Void message()

{

System.out.println("Inside parent class");

```

}

}

Class Derived extends Parent

{

Void message(){

System.out.println("inside derived class");

}

Void display()

{

Message();

Super message(); //Line1

}

}

Class SuperDemo

{

Public static void main (String args[])

{

Derived derived=new Derived();

Derived display(); //line2

}

}

```

1. Inside parent class  
Inside derived class
2. Inside derived class
3. Inside parent class
4. Inside derived class  
Inside parent class

141. What will be the output of below code snippet?

```
Package com.infy;
Class Pet{
 Public void displayName(){
 System.out.println("Inside Pet");
 }
}
```

```
Package java.pack1;
Import com.infy.pet;
Public class Dog extends pet{
 Public void displayName(){
 System.out.println("Inside Dog");
 }
}
```

```
Package java pack1;
Import com.infy.pet;
Public class Demo{
 Public static void main (String args[]){
 Pet pet=new Dog();
 Pet.displayName();
 }
}
```

1.inside Pet

2.Inside Dog

3.inside Pet Inside Dog

4.Compilation fails

142. Which of the below code is implemented without best practices standard ?

i.       List list;

    Public List getList{

        If(list.size()==0)

            Return null;

        Else

            Return list;

    }

ii.       Integer i1=new Integer(11);

    Integer i2=new Integer(11);

    System.out.println(i1==i2);

iii.      String[] str=new String[]{"Hi","Hello","Welcome"};

    List strList=Arrays.asList(str);

    For(iterator itr=strList.iterator();itr.hasNext();){

        System.out.println(itr.next());

    }

1.Option(i) is valid

2.Option(ii) is valid

3.Option(ii) and (iii) are valid

4.option(i) and (ii) are valid

143. What is Magic Number in Java in the context of Java programming best Practices?

1.A number which gets printed on the console

2.A direct usage of the number in the code

3.A number which magically disappears from the code

#### 4.A number which is generated through error

17.Predict the output of the below code

```
Public class Demo{
 Static int x=232;
 Int y=135;

 Public void display(){
 System.out.print("Inside Demo");
 }
 Public static void staticMethod(){
 System.out.print(x)//Line6
 }
 Public static void main(String[] a){
 Demo staticMethod();//Line13
 Demo demo=new Demo();
 Demo staticMethod()//Line15
 staticMethod();
 demo.display()//Line16
```

- a) 2332 135 Inside Demo 232 135 inside Demo232 135 inside Demo
- b) 232232232inside Demo
- c) 232232inside Demo
- d) 232232

```
Public class Demo{

 Public static void main(String args[]){

 List arrList=new ArrayList();

 arrList.add("First");

 arrList.add("Second");

 arrList.add(23);

 for(String str.arrList);
 System.out.println(str);
 }
}
```

Select thee suitable code to be placed inside main method for getting the required output:

A) List arrList=new ArrayList();  
arrList.add("First");  
arrList.add("Second");

```
arrList.add(23);

for(String str.arrList);

System.out.println(str);
```

B) List arrList=new ArrayList();  
arrList.add("First");  
arrList.add("Second");

```
arrList.add(23);

for(String str.arrList);

System.out.println(str);
```

c) List arrList=new ArrayList();  
arrList.add("First");  
arrList.add("Second");

```
arrList.add("23");

for(String str.arrList);

System.out.println(str);
```

d) List arrList=new ArrayList();  
arrList.add("First");  
arrList.add("Second");

```
arrList.add("23");

for(String str.arrList);

System.out.println(str);
```

144. //Assume that the first two of three test cases fail in "Testclass"

// Assme all the required import statements are added

```
Public class testrunner{
```

```

Public static void main(String [] args){

Result result = junitcore.runclasses(testclass.class)

For (Failure failure : result.getfailures()){

System.out.println(result.wassuccessful());

}

}

1) False
2) True
3) False false true
4) False false false

```

145. Consider the below code snippet

```

Locale locale = new Locale("da","DK");

NumberFormat numberFormat = NumberFormat.getInstance(Locale);

String number = numberformat.format(100.99);

System.out.println(number);

```

Here NumberFormat.getInstance() follows which design pattern ?

- 1) Factory method pattern
- 2) Singleton pattern
- 3) Abstract Factory Pattern
- 4) Builder pattern

146. Select the valid code fragment according to Java coding standard?

- (i) Public void draw(String s){
 If(s.equals("Square")){
 drawSquare();
 }
 If(s.equals("Rectangle")){
 drawRectangle ();
 }
 }

```
(ii) Public void draw(String s){
 If("Square".equals(s)){
 drawSquare();
 }
 If("Rectangle".equals(s)){
 drawRectangle();
 }
 }
```

- 1.only option (i) is valid
2. only option (ii) is valid
- 3.Both (i) and (ii) are valid
4. Both (i) and (ii) are invalid

147. What will happen to the following code when trying to get compiled?

Class RepeatableAnnotation

```
{
 @SuppressWarnings("all")//line 1
 @SuppressWarnings("deprecation")//line 2
 Public void over()
 {
 New Date(). setDate(00);
 }
}
```

- 1.Unreachable code error will be generated at line 2
- 2.Compilation will not be successful as @SuppressWarnings annotation is non-repeatable in nature
- 3.warnig will be issued as it is totally unnecessary to mention @SuppressWarnings("deprecation")
- 4.code will get complied successfully with out any warning

148. What is true regarding the following code snippet?

```

Interface StaticInterface

{
 Static void staticMethod()

 {
 System.out.println("inside interface");
 }
}

Class StaticInterfaceImpl implements StaticInterface

{
 Public void staticMethod()

 {
 System.out.println("inside interface");
 }
}

```

```

Public class StaticDemo

{
 Public static void main(String[] args)

 {
 New StaticInterfaceImpl() staticMethos();
 }
}

```

1. code will not get complied as the static method should always be public
2. code will not get complied as the static method is overridden in StaticInterfaceImpl
- 3. code will print “inside interface” on execution**
4. code will print “inside class” on execution

149. What is the result of attempting to compile and run this program?

```
Public clas collectionsDemo{
```

```
Public static void main(String argv[]){
 ArrayList arrList=new ArrayList();
 ArrayList arrListStr=arrList;
 ArrayList arrListBuf=arrList;
 arrListStr.add(1,"SimpleString");//line6
 StringBuffer strBuff=arrListBuf.get(0);//line7
 System.out.println(strBuff.toString());//line8
}
}

1.simpleString
2.compilation fails because of an error in line6 and line8
3.compilation fails because of an error in line 7
4.null
```

150. What will be the output of the following code?

```
Public class WrapperClassDemo{
 Public static void main(string aa[])
 {
 Integer intWrapper=Integer.valueOf("12345");
 Integer intWrapper2=Integer.valueOf("11",2);
 Integer intWrapper3=Integer.valueOf("E",16);
 System.out.println(intWrapper+" "+ intWrapper2+" "+ intWrapper3);
 }
}
```

1.12345 13 14  
2.12345 11 14  
3.12345 3 14  
4.12345 3 15

151. What is the result if we compile and execute the below code?

```
Public class StringTest{
 Public static void main(String[] args){
 String joinString=String.join(".", "java", "programming", "course");
 String s1="JAVA", s2="java", s3="Java";
 S1.toLowerCase();
 S3=s3.replace("J", "j");
 System.out.println(joinString);
 System.out.println(s1.equals(s2)+" , "+(s2==s3));
 }
}
```

1.java:programming:course  
False,false

2. java:programming:course  
False,true

3. java:programming:course  
True,true

4. **java:programming:course**  
**False,false**

152. Public class TestDemo{

```
 Public static void main(sting[] args){
 String value1="Hello";
 String value2=new String("Hello");
 System.out.println(value1.equals(value2)+" , "+(value1==value2));
 String value3=value2.intern();
 System.out.println((value1==vlaue3)+" , "+value1.equals(value3));
```

```
}

}

1.false,true
True,true
2. true,false
true,false
3. true,false
True,true
4. false,true
false,true
```

### 153. Public class Demo

```
{
Public static void main(String[] args)
{
Try
{
Return;
}
Finally
{
System.out.println("finally");
}
}
}
}
1.Finally
2.compilation fails
3.the code runs with no output
4.an exception is thrown at runtime
```

154. Given

```
Public class TestDemo{
 Private static Object staticObject;
 Public static Object createStaticObject(){
 If(staticObject==null){
 staticObject=new Object();
 }
 Return staticObject;
 }
}
```

What changes are required in the above code for successful execution?

- 1.The method createStaticObject should be synchronized
- 2.The method createStaticObject should be private**
- 3.The staticObject reference should not be static
- 4.The method createStaticObject should not return Object type

155. What will happen to the following code when trying to get compiled?

```
Class RepeatableAnnotation
{
 @SuppressWarnings("all")//line1
 @SuppressWarnings("deprecation")//line2
 Public void over()
 {
 New Date(). setDate(00);
 }
}
```

- 1.Unreachable code error will be generated at line 2

- 2.Compilation will not be successful as @SuppressWarnings annotation is non-repeatable in nature
- 3.Warning will be issued as it is totally unnecessary to mention @SuppressWarnings("deprecation")
- 4.code will get compiled successfully with out any warning**

156. What will happen when the following code is subjected to compilation and execution?

```
Interface DefaultMethodInterface1{
 Default public void defaultMethod(){
 System.out.println("DefaultMethodInterface1");
 }
}

Interface DefaultMethodInterface2{
 Default public void defaultMethod(){
 System.out.println("DefaultMethodInterface2");
 }
}

Public class TestDemo implements DefaultMethodInterface1, DefaultMethodInterface2{
 Public static void main(String[] args){
 DefaultMethodInterface1 defMethIn=new TestDemo();
 defMethIn.defaultMethod();
 }
}
```

- 1.An exception is thrown at runtime
- 2.Compilation fails**
- 3.DefaultMethodInterface1 will get printed on the console
4. DefaultMethodInterface2 will get printed on the console

157. What is the true regarding the following code snippet?

Interface StaticInterface

```

{
Static void staticMethod()
{
System.out.println("inside interface");
}
}

Classs StaticInterfaceImpl implements staticInterface
{

Public void staticMethod()
{
System.out.println("inside class");
}
}

Public class statiDemo
{

Public static void main(String[] args)
{
New StaticInterfaceImpl().staticMethod();
}
}

1.code will not get compiled as the static method should always be public
2.code will not get compiled as the static method is overridden in StaticInterfaceImpl
3.code will print :inside interface" on execution
4.code will print "inside class" on execution

```

158. What happens if “default” keyword is omitted while defining a default method in interface?

```

Interface interface1
{
Void method1()

```

```
{
 System.out.println("inside default method");
}
}

a.the method cannot be overridden in the implementing classes
b.the method can be overridden in the implementing classes
c.the method cannot be given body in the interface
d.compilation error occurs

1.a and b
2.a,b and c
3.c and d
4.b and c
```

159. What will happen when the following code is compiled and executed?

```
Public class TestDemo
{
 Public static void main(string[] args)
 {
 LocalDate date=LocalDate of(12,11,2017);
 System.out.print(date);
 }
}
```

1.12 11 2017 will get printed  
2.11 12 2017 will get printed  
3.compilation error will be raised as the date component is not in range  
4.Execution will get raised as the date component is not in range

160. Predict the output for the below code snippet?

```
Public class TestDemo{
 Public static collection get(){
 Collection sorted =new LinkedList();
 Sorted.add("B");
 Sorted.add("C");
 Sorted.add("A");
 Return sorted;
 }
}
```

```
Public static void main (String[] args){
```

```
 For(Object obj:get()){
```

```
 Systemout.print(obj+",");
```

```
}
```

```
}
```

```
}
```

1.A,B,C

2.B,C,A

**3.compilation fails**

4.The code runs with n output

161. Which of the following statements are true if a duplicate element objT is added to a HashSet?

**1.The element objT is not added and add() method returns false**

2.The element objT is added successfully

3.An exception occurs during runtime

4.An exception occurs during compile time

162. Which of these statements compile?(choose at least one)

**checkbox**

**1.HashSet hs=new HashSet();**

```
2. HashSet set=new HashSet();
3.List list=new Vector();
List values=new HasgSet();
List objects=new ArrayList();
Map hm=new HashMap();
```

163. What will happen when the folloeing code is executed?

```
Import java.util.*;

Public class TestDemo{

Public static void main(String[] args){

List list1=new ArrayList();

List1.add("1");

List1.add("2");

List1.add("3");

List list2=new LinkedList(list1);

List1.add(list2);

List2=list1.subList(2,5);

List2.clear();

System.out.print(list1+"");

}
}
```

1.the program complies successfully and throws exception during runtime

- 2.the program has compilation error
- 3.the program compiles successfully and executes without displaying anything
- 4.the program compiles successfully and displays 1 2 3 as output

Added on 6<sup>th</sup> November

163.Given

```
Public class Fork{
```

```

Public static void main(String[] args){

If(arg.length ==1 | arg[1].equals("test")){
System.out.println("test case");
}

else{
System.out.println("production"+ args[0]);
}

}

```

What is the result when we execute the command -line invocation as: java Fork live2

- 1.test case
- 2.production live2
- 3.production
- 4.ArrayIndexOutOfBoundsException is thrown at run time**

164.What is the result when the following code is completed and executed?

```

Class Light{
 Boolean isOn;
}

Void turnOn(){
 isOn=true;
}

Void turnoff(){
 isOn=false;
}

}

Class LightDemo{

Public static void main(String[]args){

 Light light1=new Light();
 Light light2=new Light();
}

```

```
light1.turnOn();
System.out.println("light1 is on");
light1.turnOff();
System.out.println(light1.isOn);
System.out.println(light2.isOn);}}
```

i. True

False

False

j. True

False

True

k. False

False

False

l. False

False

True

Ans : True

False

Null

165, Pay attention to the following interface and predict the correct option.

Interface Benefits

{

void calculateTotalAllowance();

}

Interface AddedBenefits extend Benefits

{

Protected void computeDA();

}

1. interface definition are wrong as interfaces are not made public

2.interface Benefits is wrong as its method is not made public

3.interface AddedBenefits is wrong as its method is made protected

4.interface Benefits is wrong as its method misses the keyword abstract

166.Given

Class ArrayDemo

```
{
 Public static void main(String[] arg){
 Int x[]={};
 for(int i=0; i < x.length; i++)
 System.out.println(x[i]+”);
 }
 Public static int[] display()
 {
 //Line 1
 }
}
```

Which code fragment can be inserted at Line 1 to print the output as 112 142 213?

- 1.new int[]{112,142,213};
2. new int{112,142,213};
3. return new int[{112,142,213}];
4. **return new int[]{112,142,213};**

167.Which of the below 'if' statement can be used to find a year is a leap year or not?

- 1.if(((year % 4 == 4) && (year % 100!= 0)) || (year % 400 == 4))
2. **if((year % 4 == 0) && (year % 100!= 0)) || (year % 400 == 0))**
3. if(((year % 4 == 0) && (year % 100!= 4)) || (year % 400 == 4))
4. if((year % 4 == 4) && (year % 100!= 0)) || (year % 400 == 0))

168. Given

```
Public class OperatorDemo{
 Public static void main(String[] args){
 Int i1=15;
 String b1 = i1>20?"Green":i1>10?"Blue":"Violet";
 System.out.println(b1);
 }
}
```

What is the result when we compile and run this program?

1.No Output

2. Green

**3.Blue**

4.Violet

169. Consider the below class and identify the extension of output file

Class Employee

```
{
 Private int x= 10;
```

```
 Public void showX();
```

```
 {
 System.out.println(x);
 }
}
```

- 1 .class**
- 2 .java
- 3 .txt
- 4 .js

170. class Operation

```
{
 Public void addition()
 {}
}
```

Class AdvOperations extends Operations

```

{
 Void addition() //Line 1
}
}

```

Line 1 generates compilation error. Which of the below option help to resolve this?

- A. 1. `@Override` annotation is needed
- B. 2. `public` keyword has to be added
- C. 3. `protected` keyword has to be added
- D. 4. method should be declared as private

46. 171. Given an abstract Class Customer as below:

```

public abstract class Customer
{
 public abstract String getCustomerType();
}

```

Select a Valid implementation of getCustomer Type method in another class, from the below options:

- 1) abstract class C1 extends Customer{
 public string getCustomer Type()
 { return “Premium”;
 }
 }
- 2) Customer customer = new Customer(){
 public String getCustomerType()
 { return “Premium”;
 }
 }
- 3) class C1 extends Customer{
 public String getCustomerType()
 { return “Premium”;
 }
 }
- 4) new Customer(){
 public String getCustomerType()
 { return “Premium”;

```
 }
}
```

Ans : 3) class C1 extends Customer{

```
 public String getCustomerType()
 { return "Premium";
 }
}
```

172. public class TestDemo{

```
 Public void main(int x){
 System.out.println("Main 1");
 }

 Public static void main(String args[]){
 System.out.println("Hello Main");
 }
```

1.Main 1

    Hello Main

2.Hello Main

    Main

3.Main 1

4.Hello Main

173. What is the output of the following code?

Given:

```
Public class Main{

 Public static void main(String args[]){

 Int towD[][]= new int[4][]; Line 1;
 towD[0]= new int[1];
 towD[1]= new int[2];
 towD[2]= new int[3];
```

```

towD[3]= new int[4];
for(int i=0;i < 4 ; i ++){
 for(int j = 0; j < 1; j ++){
 towD[i][j]; //Line 2
 }
 System.out.println("executed");
}
}

```

1. Compilation error in Line 2 as there is no assignment operator.

2. Compilation error in Line 1 as second index of array is not provided.

3. Code gets executed and displays output as expected.

4. Code gets executed but displays no output.

174. Which of the following statement(s) regarding an abstract class is/are true in java?

I. Object of an abstract class can't be created

II. An abstract class is designed only to act as a base class in hierarchy to be inherited by other classes.

1. Only I

2. Only II

3. Both I and II

4. Neither I and II

175. Predict the output of the below code:

```

public class TestDemo{
public static void main(String args[]){
 for(int a=0 ; a < 6; ++a){
 try{
 if(a % 3 == 0)
 throw new Exception("Except1");
 try{

```

```

if (a % 3 == 1)
 throw new Exception("Except2");
System.out.println(a);
}catch(Exception inside){
 a *=2;
}finally {
 ++a;
}
}catch (Exception outside){
 a+=3;
}finally{
 ++a;
}
}
}
}

```

Options : 1-5

**Ans : 5**

176. What is the output of the following code?

```

public class Parent{
 private int display1(int i) {
 return ++i;
 }
 public int display2(int i) {
 return display1(--i);
 }
}

class Child extend Parent {
 int display1(int i); // Line 1

```

```

{
 return display2(++i); // Line 2
}
}

public class Test{
 public static void main (String[] args){
 System.out.println("Value is "+ new Child().display1(564));
 }
}

```

**Doubt – If the code executed properly 565 will be the answer. But there are two independent classes, inheritance is not implemented so there will be compilation error.**

**1.Value is 565**

177. What will be the output of below code snippet?

```

Package com.infy;

Class Pet{
 Public void displayName(){
 System.out.println("Inside Pet");
 }
}

```

```

Package java.pack1;

Import com.infy.pet;

Public class Dog extends pet{
 Public void displayName(){
 System.out.println("Inside Dog");
 }
}

```

}

```
Package java.pack1;
Import com.infy.pet;
Public class Demo{
Public static void main (String args[]){
Pet pet=new Dog();
Pet.displayName();
}
}
```

1.inside Pet

2.Inside Dog [to be the answer if there is a public class named “pet” in the code]

3.inside Pet Inside Dog

4.Compilation fails [per gova, if inside dog has to be the answer, there should be a public class pet in code]

178. Which of the following Matchers method is used to test the size of a list in a test condition in Junit?

1.is()

2.length()

3.size()

4.hashitems()

179. At which position should Varargs be placed in a parameterized method?

1.first place

2.last place

3.Second last place

4.Can be anywhere

180.Which of the following are FALSE?

(Choose 2 option)

- 1,An interface can extend from only one interface.
- 2.A class can extend from another class and at the same time implement any number of interface.
- 3.a class can extend multiple abstract classes.
- 4.many classes can implement the same interface.

181.Given

Class Message{

```
Public static void main(String args[]) throws Exception{
Integer[][] val={{10,12,14},{null},{18,20,22}};
System.out.println("Array of = " +val[1][1].intValue());
}
}
```

Find the Exception that occurs?

- 1.StringIndexOutOfBoundsException: change the val data type to String
- 2.RunTimeException: due to inappropriate use of null
- 3.ArrayIndexOutOfBoundsException
- 4.NullPointerException : in the index val[1][] null in initialiced

182.What is the result when the code is complied.

```
public class App{
public static void display(String name)
{
If(name.equals("null")){
System.out.println("Greetings");
}}
public static void main(String arg[]){
display(null);
}
```

}

**Ans – Code will compile successfully, but run time exception will be thrown as NullPointer Exception**

1.Code compiled successfully and do not print anything in the console

2.Code will compile successfully and print “Greetings”.

183. class Application{

```
public static double appList(List<? Extends Number>list) {
```

```
 double sum= 0.0;
```

```
 for(Number n : list)
```

```
 sum + =n.doubleValue();
```

```
 return sum;
```

```
}
```

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

```
 List<Integer> ref = Arrays.asList(8,1,3);
```

```
 System.out.println(appList(ref));
```

```
}
```

```
}
```

Predict the output?

**1.12.0**

2.12.0000000

3.12.0d

4.12

184.What is the output of the following code when executed?

```
public class StringTester{
```

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

```
 String name = new String (“Joeden”);
```

```
 StringBuffer newName = new StringBuffer(name);
```

```
 System.out.println(name.equals(newName));
```

```
}
```

```
}
```

**1.false**

2.true

3.NullPointer exception

4.Compilation Error

185.Which of the following class/interface is not part of the Collection family?

1.Queue

**2.Enqueue**

3.Deque

4.ArrayDeque

186.What does 'this' keyword represent in java program?

1.Object

**2.Reference Variable**

3.Method

4.Constructor

187.Which of the below statement are true about design patterns

a.These are only three design pattern defined in java

b.We can use each pattern only once per application

c.Design pattern are conceptual reusable solutions

1.Statement a,b,c

2. Statement b,c

3. Statement b

**4. Statement c**

188.How can you find the different between two dates?

1.date1.difference(date2);

2.date1.until(date2.TemporalUnit)

3.date1.between(date2)

**4.date1.minus(date2)**

189.Which of the following keyword is used to propagate an exception to its calling environment?

- 1.rise
- 2.throws
- 3.catch
- 4.thrown

190.What is magic number in java in the context of java programming best practice?

- 1.A number which gets printed in the console
- 2.A direct usage of the number in the code
- 3.A number which magically disappears
- 4.A number which is generated through error

191.Which of the below method can be defined in a single class in java, select most suitable option.

- a.void add(int,int)
- b.void add(int,int,int)
- c.int add(int,int)
- 1.a and c together
- 2.a and b together
- 3.b and c together
- 4.All option

Will be given check boxes. Answers – 2 & 3

---

192. Which of the following data structure is used by varargs in java

- a. LinkedList
- b. Array
- c. ArrayList
- d. Vector

193. Which of the following is not a valid Polymorphism in Java

- a. Method Overloading

- b. Method Overriding
- c. Constructor Overloading
- d. Constructor Overriding.

194. Class Expression

```
{
Public void calc() {
Double x = 10;
Int y = 20;
Float z = 30;
//line1
b=x+y+z;
}
}
```

Identify the suitable datatype to be used for variable b at Line1

- a. Double
- b. Float
- c. Integer
- d. DoubleFloat

195. What will be the output of the following code

```
Public class Test {
Public void method() {
For(int i=0;i<3;i++) {
System.out.println(i);
}
System.out.println(i);
}
}
```

- a. 0123
- b. 012
- c. Compile error as i is not declared

d. <one more option>

196. Annie and Jacklin are working on a java project. Annie is working on a windows machine and Jacklin on a mac. Which feature of java helps Annie and Jacklin's project to execute each others machine in different environments.

- a. Multithreading
- b. Object Oriented
- c. Architecture Neutral and Portable
- d. Memory Management.

197. Which of the following is the correct syntax to declare the abstract method evaluate with a varargs parameter called marks.

- a. public abstract double evaluate(double...marks int rollNo)
- b. public abstract void evaluate (integer rollNo Float...marks)
- c. public abstract float evaluate (integer rollNo Double...marks)
- d. Varargs cannot be used as a formal parameter in abstract methods.

198. Which of the following keywords can be used to restrict a class to be inherited in java.

- a. Abstract
- b. Final
- c. constant
- d. Private

199. public class innerclassDemo{

```
 Private int bookid = 110;
 Class Book
 {
 private int bookid = 231;
 private int getBookid()
 {
 Return bookid;
 }
 }
 Private int getBookid()
 {
 InnerclassDemo InnerClassDemo = new InnerClassDemo();
```

```

//line1

System.out.println(innerClassDemo.getbookid , getbookid);

}

}

```

200. which of the following are valid functions used to read values from the user using scanner class (choose 3)

- a. **nextInt()**
- b. **nextChar()** --- string or chars are taken directly through next() method.
- c. **nextLong()**
- d. **nextLine()**

---- Code snippets – PDF -----

## Code snippets

### Table of Contents

|                                               |     |
|-----------------------------------------------|-----|
| Sections:.....                                | 127 |
| Interface:.....                               | 127 |
| Objects: .....                                | 129 |
| Date and Time:.....                           | 129 |
| Annotations:.....                             | 130 |
| Arrays: .....                                 | 131 |
| Collections:.....                             | 133 |
| Assertion:.....                               | 137 |
| Exceptions Handling:.....                     | 138 |
| String Functions: .....                       | 141 |
| Child class, super keyword and Extends: ..... | 141 |
| Pattern: .....                                | 149 |

|                    |     |
|--------------------|-----|
| Operations: .....  | 150 |
| Sorting:.....      | 154 |
| Static Block:..... | 154 |
| Unit Test:.....    | 155 |
| Constructor: ..... | 155 |
| Regex:.....        | 156 |

## Sections:

### Interface:

201. Consider the below code snippet:

```

interface Student {
 int student_id=101;
}

class Studentimpl implements Student{
 void getStudentId() {
 student_id=102;
 }
}

```

**Compilation failed**  
file.java:7: error: cannot assign a  
value to final variable student\_id  
student\_id=102;

What will happen when the above code compiles?

- a) Compilation Error as student\_id is static field and hence we cant change its value after initialization.
- b) The code will compile successfully.
- c) The code will compile successfully but when executed , it will lead to runtime exception as student\_id field cannot be changed.
- d) Compilation error as student\_id will not be visible in StudentImpl

**Answer : A**

202. What is the output of the below code snippet?

Interface ParentInterface

```

{
 Default void display()
 {
 System.out.println("Inside default method");
 }
}
```

Interface childInterface

```

{
 Default void display()
 {
 System.out.println("Default method redefined");
 }
}
```

Public class Demo implements ParentInterface, childInterface{

    Public static void main(String[] args)

```

 {
 Demo demo = new Demo();
 Demo.display();
 }
}

```

<<<No Options>>>

Answer:

Duplicate default method display from parent AND child interface

Objects:

203) Consider the below code snippet:

```

public class TestDemo {

 public static void main(String[] args) {
 // TODO Auto-generated method stub
 int i=4;
 int j=4;
 System.out.println(i==j);
 Integer k=new Integer(4);
 Integer l=new Integer(4);
 System.out.println(k==l);
 }
}

```

What would be the issue in the above code?

True False

- a) Comparing Wrapper classes objects using == is wrong as it checks if both points to same object
- b) Comparing primitive data types using == is wrong
- c) Both A and B
- d) No issues in the above code

Answer : A

Date and Time:

204) Which will be the output of the code given below?

```

public static void main(String[] args){
 LocalDateTime date1 = LocalDateTime.now();
 System.out.println(date1.plus(Period.ofDays(-1)));
}

```

2020-08-03T14:27:42.015

- a) Yesterday's Date and Time
- b) Error as LocalDateTime.now() method is not defined
- c) Will print nothing as date1.plus() method has void as its return type
- d) Error as Period.ofDays() method only take positive values

**Answer : A**

205) Given

```
public class Sample {
 public static void main(String[] args) {
 LocalDateTime dateTime = LocalDateTime.of(2020,06,7,1,1).plusHours(30);
 DateTimeFormatter formatterPattern = DateTimeFormatter.ofPattern("dd-MMM-yyyy");
 String str = dateTime.format(formatterPattern);
 System.out.println(str);
 }
}
```

Predict the output?

- a) 08 -06-2020
- b) 07-Jun-2020
- c) 08-Jun-2020
- d) 09-06-2020

**Answer : C**

206)

```
LocalDateTime localDateTime=LocalDateTime.of(2020, 5,13,20,46);
System.out.println(localDateTime.get(ChronoField.HOUR_OF_DAY)+localDateTime.getDayOfMonth());
Ans: 33
```

Annotations:

207) Refer the below code snippet and predict the outcome:

```
import java.lang.annotation.Repeatable;
import java.lang.annotation.Retention;
import java.lang.annotation.RetentionPolicy;
class annotationdemo {
```

```

@Retention(RetentionPolicy.RUNTIME)
public @interface Chocolates{
 Favourite[] value() default{};
}

@Repeatable(value = Chocolates.class)
public @interface Favourite{
 String value();
}

@Favourite("Dairy Milk")
@Favourite("Kitkat")
@Favourite("5 Star")
@Favourite("Galaxy")

public interface Chocolate{
}

```

Dairy Milk  
Kitkat  
5 Star  
Galaxy

```

public static void main(String args[]){
 Chocolates Chocolate = Chocolate.class.getAnnotation(Chocolates.class);
 for(Favourite favourite:Chocolate.value()){
 System.out.println(favourite.value());
 }
}

```

### Arrays:

208) Given

```

public class ArraysDemo{
 public static void main(String[] args){
 int[] arrVar={11,22,33,44,55,66,77,88,99,109};
 int position=3;
 int value=7;
 System.out.println("Original Array : "+Arrays.toString(arrVar));
 }
}

```

```

for(int i=arrVar.length-1;i>position;i--){
 arrVar[i]=arrVar[i-1];
}
arrVar[position]=value;
System.out.println("New Array: "+Arrays.toString(arrVar));
}

```

Identify the outcome of the given code snippet

**ANSWER :**

New Array: [11, 22, 33, 7, 44, 55, 66, 77, 88, 99]

Original Array : [11, 22, 33, 44, 55, 66, 77, 88, 99, 109]  
New Array: [11, 22, 33, 7, 44, 55, 66, 77, 88, 99]

209) class Demo1 {  
public static void main(String[] args) {  
int i=0;  
int[] myArray = {11,111};  
for(int i : myArray) { //Line1  
System.out.println("%d",i);  
}}}

- a) Primitive types cant be used in an advance for loop.
- b) Variable i cannot be resolved to a type
- c) Duplicate Local Variable i
- d) Type mismatch: cannot convert from element type int to i

**Answer : c**

210) what is the output of the following code?

Given:

```

public class Main {
 public static void main(String[] args) {
 int twoD[][]=int int[4][]; //Line1
 twoD[0] = new int[1];
 twoD[1] = new int[2];
 twoD[2] = new int[3];
 }
}

```

```

twoD[3] = new int[4];

for(int i = 0;i<4;i++){
for(int j=0;j<i+1;j++){
twoD[i][j]; //Line 2
}

System.out.println("executed");}
}

```

**Answer :—~~Executed~~ compilation error at line 1,2-**

211) What is the output of the following code?

```

Public class Demo11{

Public static void main(String args[]){

Int x[][] = new int[4][]; //Line1

X[0] = new int[1];

X[1] = new int[2];

X[2] = new int[3];

X[3] = new int[4];
}

```

```

Int a,b,c = 0;

For(a=0;a<4;a++){

For(b=0;b<a+1;b++){

X[a][b]=c;

System.out.print(x[a][b] +" ");
}
}

```

**Ans:**

**0 1 2 3 4 5 6 7 8 9**

Collections:

212) int[] myArray = new int[] {1,2,3,4,5}

Which code snippet given below allow you to create a list from this array?

- a) List myList = myArray.asList();

- b) List myList = Arrays.asList(myArray);
- c) List myList = new ArrayList(myArray);
- d) List myList = Collections.fromArray(myArray);

Answer: B

213) Consider the below code snippet

```
String[] customers = new String[]{"John", "Jack", "Jacklin", "Jasmine"};
List customersList = Arrays.asList(customers);
```

Best Code to iterate through the customersList obj data

i. for(Iterator itr = customersList.iterator();itr.hasNext();){  
System.out.println(itr.next());  
}

ii. for(String s:customerList)  
System.out.println(s);

iii. for(Iterator itr = customersList.iterator();itr.next();){  
System.out.println(itr.next());  
}  
iv) for(Iterator itr=customersList.iterator(customersList.size());itr.hasPrevious();){  
System.out.printl(s.previous());  
}

- a) Option i
- b) Option ii
- c) Option iii
- d) Option iv

Answer : A

214) Assuming all the necessary imports are done , what will be the output of the following code snippet?

```
class Movie {
```

```

private String movieName;

public Movie(String Name) {
 this.movieName =name;
}

@Override

public boolean equals(Object obj) {
 return true; }

public class CollectionTester{

 public static void main(String[] args) {
 HashMap<Movie,String> haspMap = new HashMap<>();
 hashMap.put(new Movie("Harry Potter1"),"movie1");
 hashMap.put(new Movie("Harry Potter1"),"movie2");
 hashMap.put(new Movie("Harry Potter1"),"movie3");
 System.out.println(hashMap.size());
 System.out.println(hashMap.get(new movie("Harry Potter")));
 }
}

```

**Answer : 3 Null**

215) Given

```

class Task {

 public static void main(String[] args) {
 TreeSet set = new TreeSet();
 set.add("a");
 set.add("6");
 set.add("c");
 Iterator itr = (Iterator)set.iterator();
 while(itr.hasNext()){
 System.out.println(itr.next()+"");}}
}

```

Predict the Output?

- a. a 6 c – Executed in eclipse if 6 is marked as “6”
- b. a followed by exception

- c. The code will give Compile time error cannot add String
- d. The code will java.lang.ClassCastException incompatible with java.lang.Integer :- set is defined as Generic (TreeSet instead of TreeSet<String>) means it can accept anyType value. First value is added as String so all other values should be added as String but here second value is added as int, it will not give compilation error as the set is Generic but during Runtime will give ClassCastException.

216) Assuming all the necessary imports are done, what will the output when the below code gets executed?

```
Public static Iterator getIterator(List list){
 Collection.rotate(list,1);
 Collection.reverse(list);
 Return.list.iterator();
}

Public static void main(String[] args){
 List list = new ArrayList<>();
 List.add(404);
 List.add(390);
 List.add(503);
 Iterator iterator=getIterator(list);
 While(iterator.hasNext())
 System.out.println(iterator.next());
}
```

Ans:

Option A:

390

404

503

217) Predict the output of the below code snippet?

```
ArrayList list = new ArrayList();
```

```
List.add("Infosys");
List.add("Google");
For(String s:list){
System.out.print(" "+s);
List.clear();
}
```

Option A: It prints Infosys

Option B: Compilation fails as the line "for(String s:list)" cannot convert from elementtype

Ans: Option B

Assertion:

218) Given

```
public class AppTest {
String message ="Hello";
int length = message.length();
@Test
public void testOne(){
System.out.print(length + " ");
assertEquals(length,5);
}
@Test
public void testTwo(){
System.out.print(length + " ");
assertEquals(length,5);
}
@After
public void teardown() {
length = length +1;
System.out.print(length +" ");}}
```

```
}
```

What is the result?

Answer : Both test will pass and print 5 6 5 6 in the console

219) Predict the output of the following code?

//Assume all the required imports are added

```
public class TestDemo{
```

```
@Test
```

```
public void test() {
```

```
String a = “ ”;
```

```
Assert.assertNotNull(a);}}
```

- a) Test Passes
- b) Test fails
- c) An exception is thrown at runtime
- d) Compilation fails

Answer : A

220) //Assume all the required imports are added

```
public class TestDemo{
```

```
String a1[] = { “one”, “Two”, “three” };
```

```
String a2[] = { “one”, “Two”, “three” };
```

```
@Test
```

```
public void test(){
```

```
 // Line 1
```

```
}
```

```
}
```

Choose the wrong option?

Answer: If we place Assert.assertSame(a1,a2): at Line1 the test case will pass as it verifies the contents

### Exceptions Handling:

221) What is the output for the below code?

```
//Myexception is custom exception class
```

```

public class TestDemo{
 static void myCode() throws MyException {
 try{
 throw new MyException("TestException");
 } catch (Error | Exception ex) {
 System.out.print("Inside Error and Exception");}
 public static void main(String[] args) throws MyException {
 try {
 myCode();
 } catch(Exception ex){
 System.out.print("Inside Exception");}}
 }
}

```

a) prints "Inside Error and Exception"  
b) An exception is thrown at runtime  
c) Prints "inside Exception"

**Answer : A**

222) what is the output of the following code?

package exceptions;

```

public class ExceptionDemo {

 static class Car implements AutoCloseable {
 public void close() {
 System.out.print("Automatic Door close");
 }
 }
 public static void main(String[] args){
 try{
 Car car = new Car();
 System.out.print("inside try block");
 }

 catch(Exception e){
 System.out.print("Inside catch block");
 }
 finally{
 System.out.print("finally");
 }
 }
}

```

Answer: Inside try block finally

223) Given

```
public class Sample {
 public static void main(String[] args) throws Exception {
 try {
 System.out.println("In try block");
 System.exit(0);
 } catch(Exception ex){
 System.out.println("In catch block");
 ex.printStackTrace();
 } finally{
 System.out.println("In finally block");
 }
 }
}
```

Predict the Output

- a) In try block In finally block
- b) In try block

Answer : B

224) What is the result when the following code snippet is compiled?

```
class Tester {
 public static void main(String[] args) throws Exception {
 try {
 throw new IOException(); //Line1
 } catch(IOException | Exception e) { //Line2
 System.out.println(e + "caught"); //Line3
 }
 }
}
```

- a) The code will compile and prints IOException caught
- b) The code will not compile due to error in Line 2 i.e. IOException already caught.
- c) The code will not compile and shows compile time error in Line 2 and Line 3
- d) The code will compile and prints java.IOException caught

## String Functions:

```
225) public class Util {
 public static void main(String[] args){
 String name = "Martin";
 Name.concat="Luther";
 System.out.println(name.concat(" King"));}
}
```

Predict the output:

- a) Martin
- b) Martin Luther King
- c) Martin Luther
- d) Martin King

Answer : D

## Child class, super keyword and Extends:

226) Analyze the below code and predict the outcome when compiled and executed?

```
public class Demo extends Book {
 int bookid=4567;
 public int getValue() {
 return bookid;
 }
 public void call() {
 System.out.print(super.getValue()); //Line 1
 }
 Public static void main(String[] args) {
 Book book = new Book();
 Super.call(); //Line2
 }
 class Book {
 int bookid =17897;
 public int getValue() {
```

```
return bookid;}}
```

Answer :**Compilation error at line 2**

227) What is the output of the following code?

```
class Employee {
void display(char c) {
System.out.println("Employee name starts with : "+c+");
System.out.println("His Experience is 11 years")};
class Main extends Employee {
void display(char c) {
super.display(c);
System.out.println("Another employee name also starts with: "+c+");
new Employee().display("D");
display(7);
}
String display(int c){
System.out.println("His experience is : +c" years);
return "Bye";};}
```

**No Answer**

**added code:**

228) class Parent {

```
void display(char c) {
System.out.println("Employee name starts with : "+c+" ");
System.out.println("His Experience is 11 years");
}
```

```
}
```

public class ClassDemo2 extends Parent {

```
void display(char c) {
```

```
super.display(c);

System.out.println("Another employee name also starts with: "+c+" ");

new Parent().display('D');

display(7);

}

String display(int c){

System.out.println("His experience is : "+c+" years");

return "Bye";

}

public static void main(String []args) {

 ClassDemo2 obj=new ClassDemo2();

 obj.display('e');

 System.out.println(obj.display(4));

}

}
```

Option A:

His experience is: 7 Years

His experience is: 11 years

Option B:

Employee name starts with: S

His experience is: 11 years

Another employee name also starts with: S

His experience is: 7 years

Option C:

Employee name starts with: S

His experience is: 11 years

Another employee name also starts with: S Employee

His experience is: 11 years

His experience is: 7 years

Option D:

Employee name starts with: S

His experience is: 11 years

Another employee name also starts with: S

Employee name starts with: D

Ans:

Employee name starts with : e

His Experience is 11 years

Another employee name also starts with: e

Employee name starts with : D

His Experience is 11 years

His experience is : 7 years

His experience is : 4 years

Bye

229) 3) what is wrong with respect to the code that is given below

```
class Student {
 protected static String mailDomain = "@infosys.com";
 //instance variables and methods follow
}

class Hostelite extends Student{
 public void generatedReport(String name)
 {
 System.out.println(name+Student.mailDomain);}
}
```

- a) The code will not compile as the static variables of a class cannot be used inside the instance specific methods of the child class.
- b) The Code will compile but will not get executed as the static variables of a class cannot be used inside the instance specific methods of the child class
- c) The code seems to be perfect and will work as expected

- d) The code will not compile as the keyword, implements is not used for inheriting from the parent class

**Answer : C**

```
230) 4) 39)public class Project{
 Private Integer projectId;
 Private String projectName;
 Public static void main(String[] args){
 Project oldProject=null;
 Project newProject=null;
 oldProject=new Project();
 newProject=new Project();
 newProject=oldProject;
 oldProject=new Project();
 }
}
```

Which of the following statement is true after the code gets executed?

- a) oldProject and newProject are referring to different objects.
- b) oldProject and newProject are referring to same objects.
- c) oldProject is referring to two different objects at the same time.
- d) newProject is referring to three different objects at the same time.

**Answer: A**

231) 42) what is the result when the following code is compiled and executed?

```
Class Demo{
```

```
 int x = 1;
 int y = 2;
```

```
 Demo display(Demo demoParam)
{
 Demo obj=new Demo();
 obj=demoParam;
 obj.x=demoParam.x++ + ++demoParam.y;
 demoParam.y=demoParam.y;
 return obj;
 }
```

```
 Public static void main(String[] args){
 Demo obj1=new Demo();
 Demo obj2=obj1.display(obj1);

 System.out.println("obj1.x = " + obj1.x + "obj1.y = " +obj1.y);
```

```
System.out.println("obj2.x = " + obj2.x + "obj1.y = " +obj2.y);
}
}
```

a) obj1.x=4 obj1.y=4  
    obj2.x=4 obj1.y=4

b) obj1.x=3 obj1.y=3  
    obj2.x=4 obj1.y=3

c) obj1.x=4 obj1.y=3  
    obj2.x=4 obj1.y=3

d) obj1.x=3 obj1.y=4  
    obj2.x=4 obj1.y=3

Answer: No Answer

obj1.x = 4obj1.y = 3  
obj2.x = 4obj1.y = 3

232) 57) What is the result of the following code?

```
Public class Vehicle{
 Static class Car{
 Public void go(){
 System.out.println("Car Ignition");
 }
 }

 Static class ElectricCar extends Car{
 Public void go(){
 System.out.println("ElectricCar Ignition");
 }
 }

 Static class PetrolCar extends Car{
 Public void go(){
 System.out.println("PetrolCar Ignition");
 }
 }
}
```

```
Public static void main(String[] args){
 Car car = new ElectricCar();
 Car.go();
}
```

<<<No Options>>>

Yes, you can **declare a class static** in Java, provided the **class** is inside a top-level **class**. Such clauses are also known as nested **classes** and they can be **declared static**, but if you are thinking to make a top-level **class static** in Java, then it's not **allowed**

233) 65) What is the result when the following code is compiled and executed?

```
Public class Test{
 Public void method(){
 For(int i=0;i<3;i++){
 System.out.print(i);
 }
 }

 Public static void main(String args[]){
 Method();
 }
}
```

Option A: 012

Option B: 0 1 2 3

Option C: Compilation fails as cannot make a static reference to the non static method

Option D: 2 1 0

Ans: C

234) 68) Predict the output of the below code

```
Class Car{
 Void start(){
 System.out.println("car Starts");
 }
}
```

```

 }
}

Class Bike{
 Void start(){
 System.out.println("Bike Starts");
 }
}

Class Automobile extends Car{
 Void start(){
 System.out.println("Automobile Starts");
 }
}

Public class ExceptionDemo{
 Public static void main(String[] args){
 System.out.println("implementing type casting");
 Car d = new Car();
 Automobile automobile = (Automobile)d;
 }
}

```

**Answer:**

```

implementing type casting

Exception in thread "main" java.lang.ClassCastException: snippet.Car incompatible
with snippet.Automobile
 at snippet.class68.main(class68.java:22)

```

235) 64) Given

```

Class Aircraft{
 String name = "MiG";
 String start(){
 Return "main engine Start";
 }
}

```

```

}

Class CivilianAircraft extends Aircraft{

String name = super.name;

String start(){ //Line 1

Return "Secondary engine start";

}

}

Public class Demo{

Public static void main(String[] args){

New Demo().go();

}

Void go(){

Aircraft aircraft = new CivilianAircraft(); //Line2

System.out.println(aircraft.name + "" + aircraft.start());

}

}

```

Option A: MiG Main engine start

**Option B: MiG Secondary engine start**

#### Pattern:

236) 1) What changes need to be made in the following code to make the singleton pattern correct?(Choose any 2)

```

public class Employee {

public static Employee employeeInstance;

private Employee() {}

public static Employee getEmployee()

{

if(employeeInstance==null){

employeeInstance = new Employee();

```

```
}

return employeeInstance;}}
```

- a) None of the Singleton Pattern is properly implemented
- b) Rename employeeInstance to Instance**
- c) Add synchronized to getEmployee()
- d) Change the access modifier of employeeInstance from public to private**

### Operations:

```
237) 1) public class TestDemo {

public void main(int x) {

System.out.println("Main1")

}

public static void main(String args[]){

System.out.println("Hello Main");}}
```

- a) Main1
- Hello Main
- b)Hello Main
- Main1
- c) Main1
- d)Hello Main**

**Answer : D**

238) 44) Predict the output of the following code:

```
Public Class Main{
Public void display(int i)
{
System.out.println("inside first");
}
Public void method(int i, int j)
{
System.out.println("inside second");
}
Public void method(int...k)
{
System.out.println("inside third");}
```

```
}
```

```
Public static void main(String[] args){
```

```
{
```

```
new Main().method(110);
```

```
new Main().method(110,210);
```

```
new Main().method(110,210,310); //Line1
```

```
new Main().method(110,210,310,410); //Line2
```

```
}}
```

- a) inside first  
    inside second  
    inside third
- b) inside first  
    inside second  
    inside third  
    inside third
- c) inside third  
    inside second  
    inside third
- d) inside third  
    inside second  
    inside third  
    inside third

Answer: D

239) 45) What is the result when the following code is compiled and executed?

```
Public class Test{
```

```
 Public void method(){
```

```
 for(i = 0; i<3; i++){
```

```
 System.out.print(i);
```

```
 }
```

```
}
```

```
Public static void main(String[] args){
```

```
 method();
```

```
}
```

```
}
```

- a) 012
- b) 0 1 2 3
- c) Compilation fails as cannot make a static reference to the non static method.
- d) 2 1 0

Answer: C

240) 59) What is the output for the below code snippet?

```
Public class TestDemo{
 Public static void main(String[] args){
 Try{
 Int a = 20/20;
 Int b = 20/10;
 System.out.println("a="+a+"b="+b);
 Try{
 If(a==1){
 A=a/(a-a);
 }
 If(b==2){
 Int c[]={1};
 C[22]=99;
 }
 }
 Catch(ArithmeticException ae){
 System.out.println("ArithmeticeException Bock 1");
 }catch(ArrayOutOfBoundsException ai){
 System.out.println("ArrayIndexOutOfBoundsException Block");
 }
 Catch(ArithmeticException ae){
 System.out.println("ArithmeticeException Block 2");
 }
 }
}
```

Answer:

a=1b=2  
ArithmeticeException Bock 1

241) 3)

```

public class OperatorsDemo {

 public static void main(String[] args) {
 // TODO Auto-generated method stub
 int x=120,y=110;
 String s1="Thank",s2="Thank";
 int arr1[]={1,2,3};
 int arr2[]={1,2,3};
 boolean boo=true;
 System.out.println("X==y:"+ (x==y));
 System.out.println("X<=y:"+ (x<=y));
 System.out.println("s1==s2:"+ (arr1==arr2));
 System.out.println("boo=true:"+ (boo==true));
 }
}

```

}

Ans:

X==y:false

X<=y:false

s1==s2:false

boo=true:true

242) 4) correct usage of relational operator inside if statement:

String firstName = "Annie";

int salary = 0;

- b) if(firstName=="Annie");
- c) if(firstName.equals("Annie"));
- d) if(firstName.equals("Annie")&& salary==50000);**
- e) if(firstName=="Annie" | !salary==50000);

243) 6) int x1=5;

int y1=7;

```
System.out.println(~x1+"~x1);//line1
x1&=y1;
System.out.println(x1="+x1);
```

Ans:

~x1=-6  
x1=5

### Sorting:

244) 53) Predict the output of the below code snippet?

```
Collection sorted = new LinkedList();
Sorted.add("B");
Sorted.add("C");
Sorted.add("A");
For(Object object : sorted){
 System.out.print(object +",");
}
```

Option A: A,B,C

Option B: B,C,A

Ans: Option B

### Static Block:

245) 78. What will be written at Line 1 so that the below code will compile and run successfully?

```
Public class Main {
 // line 1
 static {
 X[0]=102;
 } public static void main(String[] args) {
 System.out.println(x[0]);
 }
```

Ans : static int[] x = new int[3]

Unit Test:

```
71) public class TestDemo{
 @Before
 Public void beforeTest1(){
 System.out.printLn("in before test2");
 }
 @Before
 Public void beforeTest2(){
 System.out.println("in before test 1");
 }
 @Test
 Public void test(){
 String a = "123";
 Assert.assertSame("123" ,a);
 }
}
```

Answer:

in before test 1

in before test2

Constructor:

```
246) 1) class Greet{
 private static Greet greet=new Greet();
 private Greet() {
 }
 public static Greet getInstance() {
```

```

 return greet;
 }

 public void displayMessage() {
 System.out.println("Hey! have a great day!");
 }
}

public class Tester {

 public static void main(String[] args) {
 Greet greet=new Greet();
 greet.displayMessage();
 }
}

```

Ans:

**Compilation error: Constructor Greet() is not visible**

[Regex:](#)

```

247) 1) import java.util.regex.Matcher;

import java.util.regex.Pattern;

public class TestDemo {

 public static void main(String[] args) {
 // TODO Auto-generated method stub
 }

 try {
 checkName("TioT");
 }

}catch(Exception r) {
 System.out.println("Exception 1");
}

```

```

}

}

static void checkName(String name) {
 Pattern regex=Pattern.compile("T[aei]{3}T");
 Matcher mobileMatcher=regex.matcher(name);
 try {
 if(mobileMatcher.matches()==false) {
 throw new RuntimeException("Name is invalid");
 }
 }else {
 System.out.println("Name is valid");
 }
}catch(NullPointerException r) {
 System.out.println("Exception 2");
}
}
}

```

**Ans: Exception 1**

---- PREM ----

248) Which of the following methods can be defined in a single class in Java (choose most suitable 2 options)

- Void add(int, int)
- Void add(int, int, int)
- Int add(int, int)

- A. 1 & 3
- B. 1 & 2**
- C. 2 & 3
- D. 1,2 & 3

Will be given check boxes.

Answers – 2<sup>nd</sup> and 3<sup>rd</sup> options

249) Which of the following is the correct way to define a generic method in java (choose 3)

- A. access specifier <generic type-parameter-list> return-type method-name(parameter-list) {}**
- B. access specifier static-keyword <generic type parameter-list> return-type method-name (parameter-list) {}**
- C. access specifier return-type <generic type-parameter-list> method-name (parameter-list) {}
- D. <generic type-parameter list> return-type method-name (parameter-list) {}

250) Which of the below statements are true about design patterns?

- 1. There are only three design patterns defined for Java
- 2. We can use each design pattern only once per application
- 3. Design patterns are conceptual reusable solutions

- A. Statements 1,2,3
- B. Statements 3,2
- C. Only Statement 2
- D. Only Statement 3**

251) What does ‘this’ keyword represent in a Java program?

- A. Object
- B. Reference Variable**
- C. Method
- D. Constructor

252) How can you find the difference between two dates?

- A. date1.difference(date2):
- B. date1.until(date2.TemporalUnit):
- C. date1.between(date2):

D. date1.minus(date2);

253) Which of the following statements are FALSE?

(choose 2 options)

- A. An Interface can extend from only one interface ,
- B. A class can extend from another class and at the same time implement any number of interfaces.
- C. A class can extend multiple abstract classes ,
- D. Many classes can implement the same Interface .

254) Which of the following keywords is used to propagate method an exception to its calling environment??

- A. Raise
- B. Throws
- C. Catch
- D. thrown

----- Dec 5<sup>th</sup> 2020-----

255) Which is an incorrect example of polymorphism in Java??

- A. Method Overriding
- B. Method Overloading
- C. Constructor Overloading
- D. Constructor Overriding

256) All wrapper classes (character, Boolean, Byte, integer, double, float, Long, Short) extend which two of the below classes

- A. Serializable
- B. Number
- C. Comparable
- D. BufferedReader

257) What is false about object in Java??

- A. Object contains memory location in Heap
- B. Object contains memory location in Stack
- C. Object is an instance of a class
- D. Objects can communicate with each other

258) Which of the following doesn't support "this" keyword ??

- A. Default & parametrized constructors
- B. Static blocks & methods**
- C. Non static & private
- D. Abstract Class

259) What is false about Errors in Java exception??

- A. Serious problem that should Not be caught.
- B. Usually occurs due to lack of system resources**
- C. Errors can be handled by code in the program
- D. An error is taken as unchecked exception

260) What is the default format of date in java?

- A. Yyyy-mm-dd**
- B. mm-dd-yyyy
- C. yyyy.mm.dd
- D. yy-mm-dd

261) which is used to create Mutable strings in Java??

- A. String
- B. StringBuffer
- C. StringBuilder
- D. StringBuffer and StringBuilder**

262) Which of the following is false about Super keyword in Java

- A. Super Keyword can be used to call a parent class public constructor which is present in different package
- B. Super Keyword can be used to call a parent class private constructor which is present in same package**
- C. Super Keyword can be used to call a parent class protected constructor which is present in same package
- D. Super Keyword can be used to call a parent class protected constructor which is present in different package

263) Public Static Void Main (String[] name) throws Exception {

```
Integer [][] val = { {10,12,14}, {null}, {18,20,22}};
System.out.println("Array of =" + val[1][1].intValue());
}
```

}

Predict what exceptions occur??

- A. StringIndexOutOfBoundsException: change the val data type to String[]
- B. NullPointerException: due to inappropriate use of null
- C. ArrayOutOfBoundsException: there is no integer value at int[1][1]
- D. <one more option>

264) Which of the following is an incorrect way of declaring variable EmployeeData

assume Employee is a user defined enum

- A. Array<Employee> employeeData[] = new String[5]
- B. List<Employee> employeeData = new ArrayList()
- C. Map<Integer, String> employeeData = new LinkedHashMap<>();
- D. Set <Employee> employeeData = new HashSet<>()

265) Consider the following statements

java.util.calendar class implements factory design pattern

java.Util.ResourceBundle.getBundle implements factory design pattern

java.awt.Desktop.getDesktop() implements singleton design pattern

which of the statements are true

- A. All Statements

- B. Only Statement 1
- C. Only Statement 2
- D. Only Statement 3

266) The below code will generate compilation error. Select the possible options to avoid it.

```
Public interface Address {
 Static default void city() {
 System.out.println("default city method");
 }
}
```

- A. Illegal combination of city() only one of abstract, default or static is permitted
- B. Use public keyword in city()
- C. Define an abstract method in the interface
- D. In interface the method return type should not be void

267) Given:

```
Public class Fork {
 Public static void main(String [] args){
 If(args.length == 1| args(1).equals("test")) {
 System.out.println("test case");
 }
 else {
 system.out.println("production" + args(0));
 }
 }
}
```

What is the result when we execute the code with command line invocation java Fork live

- A. test case
- B. production live2
- C. production
- D. ArrayOutOfBoundsException is thrown at run time

268) Which of the following is false regarding "this" keyword in java??

- A. This keyword refers current class object
- B. This keyword can be used to call current class method
- C. This keyword can be used to refer current class constructors
- D. This keyword can be used to create a block of code

269) What is the result of the following code when compiled and executed

Class Demo

```
{
 Int x = 1;
 Int y = 2;

 Demo display (Demo demoParam)
 {
 Demo obj = new Demo();
 Obj = demoParam;
 Obj.x = demoparam.x++ + ++demoParam.y;
 DemoParam.y = demoParam.y;
 Return obj;
 }
```

```
Public static void main(String[] args)
```

```
{
 Demo obj1 = new demo();
 Demo obj2 = obj1.display(obj1);
 System.out.println("obj1.x=" + obj1.x + "obj1.y =" + obj1.y);
 System.out.println("obj2.x=" + obj2.x + "obj1.y =" + obj2.y);
}
```

a) obj1.x=4 obj1.y=4  
    obj2.x=4 obj1.y=4

b) obj1.x=3 obj1.y=3  
    obj2.x=4 obj1.y=3

c) obj1.x=4 obj1.y=3  
    obj2.x=4 obj1.y=3

d) obj1.x=3 obj1.y=4  
    obj2.x=4 obj1.y=3

executed output is

4 3

4 3

270) Which of the following is FALSE regarding polymorphism in Java??

- A. Polymorphism is the ability of an object to take different forms
- B. Polymorphism can be defined as a single action that can be performed in different ways.
- C. Polymorphism means different forms by creating different classes that are related to each other by inheritance
- D. Polymorphism is writing same method multiple times in a class with different parameters

271) Which of the following methods can be used to sort objects. Assume o1 and o2 are two instances of any object class (select two)

- A. Compare(object o1.object o2)
- B. o1.equals(object o2)
- C. o1.compareTo(object o2)
- D. o1 == o2

272) which is the correct way of placing "this" keyword in a constructor??

- A. First statement
- B. Last statement
- C. Inside a parametrized constructor only
- D. Cannot be placed in a parametrized constructor

273) What is the scope/access specifier of getEmployeeID() in the code snippet given .....

```
Class Employee {
 Int employeeID;
 Double getEmployeeID() {
 System.out.println("Employee ID");
 Return employeeID;
 }
}
```

- A. Public
- B. Private
- C. Double
- D. Default

1. Which of the following design pattern can be used to return <unknown word> which can be used to create set of related objects??
  - A. System
  - B. Factory
  - C. Abstract Factory
  - D. Prototype

274) What is the output...

```
Class Greetings {
 Public static void main(String[] args) throws Exception {
 try {
 system.out.println("Greetings!" + " " + args[0]);
 } catch (ArrayIndexOutOfBoundsException e) {
 System.out.println("Sam");
 }
 }
}
```

- A. Greetings! Sam
- B. Greetings!
- C. Sam
- D. Greetings! Args[1]

275) What can be expected when the following code is compiled??

```
Public abstract class Customer {
 Public int custID; //line1
 Customer() //line2
 {
 custID = 23456; //line3
 }
 Abstract public void SetID();
 Abstract final public void getID(); //line4
}
```

- A. Compilation error at line1 as public and int cannot be declared in abstract classes
- B. Compilation error at line2 as public constructors cannot be declared in abstract classes
- C. Compilation error at line3 as custID should be referred using *this* keyword in the constructor
- D. Compilation error at line 4 abstract methods cannot be final

276) Which of the following statements regarding an abstract calss are true in java

- I. Object of an abstract class cant be created

- II. An abstract class is designed only to act as a base class in hierarchy to be inherited by other classes
- A. Only I
  - B. Only II
  - C. Both I and II**
  - D. Neither I nor II

277) Which of the following component cannot be declared as static in java

- A. Method
- B. Variable
- C. Object
- D. Class

278) Which of the following statement is FALSE regarding switch case in java??

- A. Switch allows the flow of execution to be switched according to a value
- B. Switch is generally used as a replacement for if-else ladder
- C. Switch improves readability
- D. The value to be checked in the switch case must be non-primitive only.

279) What is the result when the following code snippet is compiled??

```
Class Tester {
 Public static void main (String[] args) throws Exception {
 Try {
 Throw new IOException(); //line1
 } catch (IOException | Exception e) //line2
 System.out.println(e + "caught") //line3
 }
}
```

- A. The code will compile and prints IOException caught
- B. The code will not compile due to error in line2 i.e. IOException is already caught
- C. The code will not compile and shows compile time error in line2 and line3
- D. The code will compile and print java.io.IOException caught

280) Which one of the following is used for the automatic accurate tracking for decimal values

- A. Float
- B. Double
- C. BigDecimal
- D. Decimal

```
281) public static void main(String args[])
{
 localDate local=LocalDate.of(2020,1,1);
 local=local.minusWeeks(-4L);
 local=local.plusMonths(-12L);
 system.out.println(local);
}
```

- A. 2019-01-29 (option says 2019-02-29 but it might be a typo. Executed code prints 29 Jan 2019 )
- B. 2020-12-04
- C. 2021-01-29
- D. 2018-12-04

282) Which one is valid upper bound by class Employee of list??

- A. List <? Super Employee>
- B. List<? Extends Employee>
- C. List(? Extends Employee)
- D. List<extends ? Employee>

283) Which of the below is not a valid classification of design pattern??

- A. Creational patterns
- B. Structural patterns
- C. Behavioral patterns
- D. Java patterns

```
284) Public class StringEqualsDemo {
 Public static void main(String[] args) {
 String name1 = new String("Java");
 String name1 = new String("Java");
 Sytem.out.println(name1==name2);
 System.out.println(name1.equals(name2));
 }
}
```

- A. True, True

- B. True, False
- C. False, False
- D. False, true

285) Assuming all necessary imports are done, what will be the output of the below code when executed

```
Public class tester {
 Public static void main(String[] args) {
 Set<integer> set1 = new HashMap<>(new Comparator<integer>() {
 @override
 Public int compare(Integer o1, integer o2) {
 Return o2.compareTo(o1);
 }
 Set1.add(234);
 Set1.add(657);
 Set1.add(143);
 System.out.println(set1);
 }
}
```

- A. {234,657,143};
- B. {143,234,657};
- C. Compilation Error: Cannot Infer Type argument from Hashmap
- D. {657,234,143}

286) Given

```
Int methodCall(int a, int b)
{
 Int c = 0;
 If ((a>10) && (b>10))
 {
 C = 10;
 }
 Return c;
}
```

Full statement coverage is ensured by which of the below functional call

- A. methodCall(5,10)
- B. methodCall(50,1)
- C. methodCall(30,9)

D. methodCall(20,20)

287) which of the following interface should be implemented by a resource, if we want to create the object of that resource in the resource part of a try block

- a. Closeable
- b. Closeable<E>
- c. AutoCloseable
- d. AutoCloseable <E>

288) Which of the following is FALSE regarding parametrized constructors in java

- A. Parametrized constructors should have void as return type
- B. Parametrized constructors can take any number of parameters
- C. Parametrized constructors cannot have private access modifier
- D. Parametrized constructors cannot throw any exception

289) How many number of values can be accommodated by the varargs in java

- A. Atleast one value
- B. Atmost two values
- C. More than one value
- D. Any number of values

All except B have very similar meaning. Not sure why C should be the right option.

290) Which of the following is not a valid polymorphism in java

- A. Method Overriding
- B. Method Overloading
- C. Constructor overloading
- D. Constructor Overriding

291) Predict the output for below code snippet

```
List list = new ArrayList();
List.add(0,"A");
List.add(0,"A");
List.add(0,"A");
For(object object: list) {
 System.out.println(" " + object);
}
```

- A. A B C

B. A C B

C. A C

D. B A C

292) What is the result of executing the following code??

```
Package exceptions;

Public class Demo {

 Public static void main(string[] args)
 {

 Try {

 Int z = 0/8

 } catch (Exception e) {

 System.out.println("Arithmetic exception:");

 } finally {

 System.out.print("Finally Block");

 }

 }

}
```

A. Arithmetic Exception Finally Block

B. Finally Block

C. Arithmetic Exception

D. Finally Block Arithmetic Exception

293) Identify the valid code to be inserted at Line 5 assume the code is running in multithreaded environment

```
Public class Demo {

 Private static string id;

 Private Random random = new Random();

 Public Demo() {

 //Line5

 Id = "ACC1101"+Math.abs(random.nextInt());

 }
}
```

```

 }

Public string getID() {
 Return id;
}

```

- A. if(id==null)
- B. synchronized(this){if(id==null){}**
- C. if(id==null){synchronized(this){}}
- D. synchronized{if(id==null){}}

294) Which of the following matchers method is used to test the size of a list in a test condition in Junit??

- A. Is()
- B. Length()
- C. Size()**
- D. Hashitems()

Pay attention to the following interfaces and predict the correct option.

#### Interface benefits

```

{
 Void calculateTotalAllowances();
}

Interface AddedBenefits extends Benefits

{
 Protected void ComputeDA();
}

```

- A. Interface definitions are wrong as interfaces are not made public
- B. Interface Benefits is wrong as its method is not made public.
- C. Interface AddedBenefits is wrong as its method is made protected**
- D. Interface benefits is wrong as its method missed the keyword abstract

295) Given

```
Public class OperatorDemo {
 Public static void main (String[] args) {
 Int i1 = 15;
 String b1=i1>20?"Green":i1>10?"Blue":"Violet";
 System.out.println(b1);
 }
}
```

What will be the result when we compile and run this program??

- A. No output
- B. Green
- C. Blue
- D. Violet

296) Which of the following is FALSE (choose 2 options)

- A. An interface can extend from only one interface
- B. A class can extend from another class and at the same time implement any number of interfaces
- C. A class can extend multiple abstract classes
- D. Many classes can implement the same Interface.

297) At which position should varargs be placed in a parametrized method??

- A. First Place
- B. Last Place
- C. Second Last place
- D. Can be anywhere

298) What is the result of the following code when compiled

```
Public class App {
 Public static void display(String name)
 {
 If(name.equals("null")) {
 System.out.println("Greetings");
 }
 Public static void main(String[] args) {
 Display(null);
 }
 }
}
```

- A. Code will compile successfully and do not print anything in the console
  - B. Code will compile successfully and prints "Greetings"
- Other options not available

Ans – Code will compile successfully, but run time exception will be thrown as NullPointerException

299) What is the output of the following code when executed

```
Public class StringTester{
 Public static void main(String[] args) {
 String name = new String("Jordan");
 StringBuffer newName = new StringBuffer(name);
 System.out.println(name.equals(newName));
 }
}
```

- A. False
- B. True
- C. NullPointerException
- D. Compilation Error

300) Public class Employee {

```
Private int empid;
Private String empName;
Private String designation;
Private transient String dob;
}
```

Analyze the given code and identify the suitable comments from below options (choose two)

- A. Fields in non-serializable classes should not be "transient"
- B. Fields in non-serializable classes should be "transient"
- C. Fields in a non-serializable class can be "transient"
- D. Make the Employee class as serializable

----- OTHERs – Could have duplicates -----

## ADVANCED JAVA

### Question set -1

#### 01\_Java\_Language\_Fundamentals

301) Question 1

What is the scope/access specifier of getEmployeeId() method M the code snippet given below?

```
class Employee
{
 int employeeId;
 double getEmployeeId(){
 System.out.println("Employee Id");
 return employeeId;
 }
}
a. Public
b. Private
c. protected
d. default If no keyword is mentioned it means the Scope is “Default”
```

302) Question 2

The below code will generate compilation error. Select the possible options to avoid it. (Choose 3)

```
class Phone{
 void phName{ //Line1
 System.out.println("Samsung");
 }
}

class Nokia extends Phone{
 void phName() throws Exception{ //Line2
 System.out.println("vivo");
 }
}

public static void main(String args[]){
 Phone obj = new Nokia();
 obj.phName();
}
```

- A. Use any UncheckedExceptions In Line2
- B. Dont throw any Exception In Line2
- C. Dont throw any checked Exceptions
- D. Use CheckedExceptions in Line1

Answer: All answer shown here is incorrect. Both Line 1 and Line 2 will show compilation error. To fix code in Line 1 add “()” after “void phName”. To fix issue in Line 2 either remove “throws Exception” in Line 2 or add “throws Exception” to “phName” method in phone class and main method in Nokia Class.

303) Question 3

Given the below code snippet, what will be the output?

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

```
float val1 = 5f;
float val2 = 2f;
float result = val1 % val2;
System.out.println(result);
}
}
```

- a. 0.0
- b. 0
- c. 1.0 - mod of val1 % val2 is one. Since it is assigned to float variable result will be 1.0
- d. 1

#### 304) Question 4

What will be the output for the below code?

```
public class Employee {
 private double salary=15000.00;

 public double getSalary(int salary){
 return salary;
 }

 public static void main(String args[]){
 Employee employee=new Employee ();
 double result= employee.getSalary(23000); // Line 8
 System.out.println("Sala ry is: "+result);
 }
}
```

- a. 15000.00 is “S” is capital letter in Salary
- b. 23000.00 int 23000 is assigned to double variable so the output is 23000.00**
- c. null
- d. 0.0

305) Question 5

Which of following is a necessary condition for implicit type conversion in Java?

- a. The destination type is smaller than source type
- b. The destination type is larger than source type**
- c. The destination type can be larger or smaller than source type
- d. The destination and the source type are the same

306) Question 6

What is the result when the following code is compiled and executed

```
public class Test
{
 Long a; //Line 1
 long b;
 public Test(long c){
 b = a+c; //Line 2
 System.out.println(b);
 }
 public static void main(String[] args){}
```

```
Test test = new Test ();
test.Test(new Long(10L));
}
}
```

- a. 10
- b. 10L
- c. **NullPointerException in Line 2 as variable "a" is not initialized** b is declared as basic datatype long but a is declared as wrapper class Long. Basic datatype will take default value if not initialized but objects of Wrapper class takes null if not initialized explicitly . So, this code will result in NullPointerException
- d. NullPointerException in Line 1 as variable "a" is not initialized

### 307) Question 7

Consider the Project class in following code snippet:

```
public class Project{
 private Integer projectId;
 private String projectName;
 public static void main(String[] args){
 Project oldProject=null;
 Project newProject=null;
 oldProject=new Project();
 newProject=new Project();
 newProject=oldProject;
 oldProject=new Project();
 }
}
```

Which of the following statement is true after the code gets executed?

- a. **oldProject and newProject are referring to different objects**
- b. oldProject and newProject are referring to same objects
- c. oldProject is referring to two different objects at the same time.

d. newProject is referring to three different objects at the same time.

308) Question 8

Given the enum definition and the Java class

```
enum Day {
```

```
SUNDAY(1), MONDAY(2), TUESDAY(3), WEDNESDAY(4), THURSDAY(5), FRIDAY(6), SATURDAY(7);
```

```
private int value;
```

```
private Day(int value) {
```

```
 this.value = value;
```

```
 public int getvalue() {
```

```
 return this.value;
```

```
}
```

```
}
```

```
public class TestEnum{
```

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

```
 for (Day day:Day.values()) {
```

```
 //Line 1
```

```
 }
```

```
 }
```

```
}
```

What should be placed at line 1 get the output as shown below?

SUNDAY-MONDAY-TUESDAY-WEDNESDAY-THURSDAY-FRIDAY-SATURDAY-

- a. `System.out.print(day.toString()+"-");` `toString` will fetch the String value of the object called.
- b. `System.out.print(day.getValue()+"-");`
- c. `System.out.print(day.names ()+"-");`
- d. `System.out.print(day.getName)+"-");`

309) Question 9

Which of the following component is responsible to compile, debug and execute a Java program?

- a. JVM
- b. **JDK**
- c. JRE
- d. JIT

**JDK** is a core component of **Java Environment** and provides all the tools, **executables** and binaries required to compile, debug and execute a Java Program.

310) Question 10

Which of the following OOP terminology is associated with the below statement?

- "An Employee has an Address"
- a. Association
  - b. **Aggregation** Has-A keyword represents Aggregation
  - c. Inheritance
  - d. Composition

### **03\_Java\_Handling\_Exceptions**

311) Question 1

What is the result when the following code snippet is compiled?

```
class Tester {
 public static void main(String[] args) throws Exception{
```

```

try{
 throw new IOException(); //Line1
} catch (IOException | Exception e){ //Line2
 System.out.println(e +"caught"); //Line3
}
}

```

- a. The code will compile and prints IOException caught
- b. The code will not compile due to error in Line2 i.e. IOException is already caught :-Exception is super class and contains IOException in it implicitly. So, not required to write IOException explicitly at Line 2
- c. The code will not compile and shows compile time error in Line2 and Line3
- d. The code will compile and prints java.io.IOException caught

### 312) Question 2

What is the result of attempting to compile and run this program?

```
class CustomException extends Exception{
```

```
}
```

```
class Customer extends CustomException {
```

```
}
```

```
public class ExceptionDemo {
```

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

```
 try{
```

```
 throw new Customer();
```

```
 }catch (CustomException customException) {
```

```
 System.out.println("CustomException catch block");
```

```
 } catch (Customer customer) {
```

```
 System.out.println("Customer catch block");
```

- ```
    }
}

}

a. Prints "CustomException catch block"
b. Prints "Customer catch block"
c. Compilation Error because Customer class exception is not throwable :- CustomException is super
class of Customer class . As the super class is already used in the first catch block the code will not reach
second catch block and results in compilation error
d. Compilation Error because Bank class exception is caught before Customer class exception
```

313) Question 3

```
public class TestDemo {
    static void myCode() throws MyException {
        try {
            throw new MyException("Test exception");
        } catch (Error | Exception ex) {
            System.out.print(" Inside Error and Exception ");
        }
    }

    public static void main(String[] args) throws MyException {
        try {
            myCode();
        } catch (Exception ex) {
            System.out.print("Inside Exception");
        }
    }
}
```

- a. prints "Inside Error and Exception"
- b. An exception is thrown at runtime
- c. prints "Inside Exception"
- d. **Compilation fails** MyException is not a built in class under Exception or Throwable. As MyException is user defined class and it's definition is not available compilation will fail

314) Question 4

Which of the following exceptions are ignored during compile time (Choose 2)

- a. **ArrayIndexOutOfBoundsException**
- b. **ClassNotFoundException**
- c. **NullPointerException**
- d. **InterruptedException**

315) Question 5

What will be the output of the program?

package exceptions;

```
public class Demo {  
    public static void main(String[] args)  
    {  
        try  
        {  
            System.out.print("try block ");  
            throw new Error();  
        }  
    }  
}
```

```
        catch(Throwable e)
        {
            System.out.print("catch block ");
        }
    finally
    {
        System.out.print("finally");
    }
}
```

- a. try block finally
- b. try block catch block
- c. try block catch block finally
- d. finally

Correct Answer: c

04_Java_Lang_And_Util_Package_classes

316) Question 1

Which of the following class is used to create an object of mutable character sequence?

- a. String
- b. StringBuffer
- c. String and StringBuffer
- d. StringBuilder

317) Question 2

Where are String objects stored in memory using "new" Keyword?

- a. In Stack Memory
- b. In String constant pool in Heap Memory**
- c. In Native Methods Stack Memory
- d. Anywhere in Heap Memory

318) Question 3

Predict the output for the below code;

```
public class TestDemo {  
    public static void main(String[] args) {  
        Integer n1 = new Integer(100);  
        Integer n2 = new Integer(100);  
        Integer n3 = 127;  
        Integer n4 = 127;  
        System.out.println(n1 == n2);  
        System.out.println(n3 == n4);  
    }  
}
```

- a. false**
- true**
- b. true
true
- c. true
false
- d. false
false

319) Question 4

Given

```
public class App {  
    public static void main(String[] args){  
        String s1= new String("smart");  
        String s2 = s1;  
        if (s1 == s2) {  
            System.out.println("==smart");  
        }  
        if (s1.equals(s2)) {  
            System.out.println("equals smart");  
        }  
    }  
}
```

Predict the output?

a. ==smart

then runtime time exception occurs

b. ==smart

equals smart

c. ==smart

d. equals smart

Correct Answer: b

Which among the following option are correct with respect to HashMap?,(Select any 2 options)

override boolean equals(Object o)

override toString()

override int hashCode()

override String hashCode()

321) Question 3

Given

```
public class Group extends TreeSet{
    public static void main(String[] args){
        Group g = new Group();
        g.add(new Person("Hans"));
        g.add(new Person("Jane"));
        g.add(new Person("Hans"));
        System.out.println("Total:" + g.size());
    }

    public boolean add(Object o){
        System.out.println("Adding:" + o);
        return super.add(o);
    }
}

class Person{
    private final String name;

    public Person(String name){
        this.name = name;
    }
}
```

```
public String toString(){  
    return name;  
}  
}
```

What will be the output when this code snippet is compiled and executed?

a. Adding: Hans

An exception is thrown at runtime as Person is not compatible with Comparable :- return super.add(o); in add method will call add method in super class TreeSet with value of Type person which will result in Runtime Exception

b. Adding: Hans

Total: 3

c. Adding: Hans

Total: 2

d. The code does not compile as Person is not compatible with Comparable

322) Question 4

Identify the Incorrect statement as per the Collection Framework hierarchy?

a. List extends Collection

b. ArrayList implements List

c. HashSet implements Set

d. LinkedHashSet implements HashSet HashSet is a class it cannot be implemented only extended.

323) Question 5

```
public class TestDemo{  
    public static void main(String[] args){  
        HashMap props = new HashMap<>();  
        props.put("key45", "some value");  
        props.put("key12", "some other value");  
        props.put("key39", "yet another value");  
        Set s = props.keySet();  
        //Line 1  
    }  
}
```

Which of the below code has to be inserted at Line 1, to sort the keys in the props HashMap variable?

- a. s = new Set(s);
- b. s = new TreeSet(s);** TreeSet will sort implicitly
- c. s = new LinkedHashSet(s);
- d. s = new SortedSet(s);

06_Java_SE8_Features

324) Question - 1

What will be the output of the following, code when executed,

```
public class DateTimeFormatterTester{  
    public static void main(String[] args){  
        DateTimeFormatter dateDateFormat= DateTimeFormatter.ISO_DATE;  
        LocalDate dob=LocalDate.of(2020,Month.FEBRUARY,31);  
        System.out.println(dateDateFormat.format(dob));  
    }  
}
```

```
 }  
 }
```

2020-02-31

2020-31-02

Compilation Error in creating DateTimeformatter reference using ISO_DATE

Runtime exception : java.time.DateTimeException with the message Invalid date 'FEBRURARY 31'

325) Question 2

What is true regarding the following code snippet?

```
interface StaticInterface  
{  
    static void staticMethod()  
    {  
        System.out.println("inside interface");  
    }  
}  
  
class StaticInterfaceImpl implements StaticInterface  
{  
    public void staticMethod()  
    {  
        System.out.println("inside class");  
    }  
}
```

```

    }
}

public class StaticDemo
{
    public static void main(String[] args)
    {
        new StaticInterfaceImpl().staticMethod();
    }
}

```

- a. Code will not get compiled as the static method should always be public
- b. Code will not get compiled as teh static method is overriden in StaticInterfaceImpl
- c. Code will print "inside interface" on execution
- d. Code will print "inside class" on execution**

Correct Answer: d

326) Question 3

What will be the output of the code given below?

```

public static void main(String[] args)
{
    LocalDateTime date1 =LocalDateTime.now();
    System.out.println(date1.plus(Period.ofDays(-1)));
}

```

- a. Yesterday's Date and Time** plus() will add days but the value provided here is minus 1 day so output is yesterday
- b. Error as LocalDateTime.now() method is not defined

- c. Will print nothing as date1.plus() method has void as its return type
- d. Error as Period.ofDays() method only takes positive values

327) Question 4

Which among the following are valid lambda expressions to sort the numbers in numberlist in descending order? (choose 3)

- a. numberList.sort((x,y)->x.compareTo(y));
- b. numberList.sort(int x, int y)-> -x.compareTo(y));
- c. numberList.sort((x,y)->{return -x.compareTo(y);});
- d. numberList.sort(Integer x, Integer y)->-x.compareTo(y));

328) Question 5

Given

```
interface Greeting{
    default void greet(){
        System.out.println("In Greet interface");
    }
}

class GreetingDef implements Greeting {
    public void greet(){
        System.out.println("In GreetingDef class");
    }
}

public class App{
```

```
public static void main(String par[]) {  
    Greeting obj = new GreetingDef();  
    obj.greet();  
}  
}
```

a. In Greet interface

In GreetingDef class

b. In GreetingDef class

c. In GreetingDef class

In Greet interface

d. In Greet interface

Correct Answer: b

07_Java_Programming_Best_Practice

329) Question 1

Which of the following is used as a better option to print all names in array and why?

String[] names = {"Alice", "Bob", "Carol", "David", "Eric", "Frank"};

i. for (int i = 0; i < names.length; i++) {
 System.out.println(names[i]);
}

```
ii. for (String name : names) {  
    System.out.println(name);  
}
```

a. Option I is better as variable i can be incremented as required. -Traversing in reverse fashion is not possible in For each

b. Option I is better as length of array is used to mention limit. – This is also fine as Limit option is not available in for each.

c. Option I is better as starting index can be changed. – This is also fine as forEach traverses through all elements

d. Option II is better as it makes code cleaner and simpler.

330) Question 2

Which of the below code is implemented without best practices standard.

```
i. String[] str=new String[]{"Hi","Hello","Welcome"};  
List strList=Arrays.asList(str);  
for(Iterator itr=strList.iterator().itr.hasNext();){  
    System.out.println(itr.next());  
}
```

```
ii. Integer i,new Integer(11);  
Integer i2=new Integer(11);  
System.out.println(i1==i2);
```

a. Option(i) doesnt follow best practices can be improved using for(String s:strList)

b. Option(ii) doesnt follow best practice as objecs should not be compared with ==

c. Option (i) and (ii) are not following best practices

d. Option (i) and (ii) are following best practices

331) Question 3

Which of the below are NOT good practice for creating objects?

a. Lazy Initialization of objects

b. Creating String literals instead of String objects

c. Creating Wrapper Objects instead of primitives

d. Invoking static factory methods for immutable classes

332) Question 4

Which of the below method name is valid as per Java naming convention?

a. METHOD_NAME

b. MethodName

c. methodName

d. Method_Name

333) Question 5

What is Magic Number in Java in the context of Java programming best practices?

a. A number which gets printed on the console

b. A direct usage of the number in the code

c. A number which magically disappears from the code

d. A number which is generated through error

08_Java_Design_Patterns

334) Question 1

Which of the following statement describes best about the Builder Pattern?

- . This pattern is used when object creation is costly

This pattern is used when we need to decouple an abstraction from its implementation so that the two can vary independently

This pattern builds a complex object using a step by step approach

This pattern refers to creation of duplicate object while keeping performance in mind b

335) Question 2

Which among the following is/are true about Design Pattern? (Select Two)

Design pattern depends upon abstraction

Design patterns are completed designs that can be transformed directly into code.

Design pattern depends on abstraction , follows the process of Dependency Injection.

Design pattern is a template of solving problem that can be used in many real world software development problems

336) Question 3

What changes need to be made in the following code to make the singleton pattern correct? (Choose any 2.)

```
public class Employee {  
    public static Employee employeeInstance;  
    private Employee (){}  
    public static Employee getEmployee(){
```

```
        if(employeeInstance == null) {  
            employeeInstance = new Employee();  
        }  
        return employeeInstance;
```

```
}
```

None. the singleton pattern is properly implemented.

Rename employee to instance.

Add synchronized to getEmployee().

Change the access modifier of employeeInstance from public to private.

09_Junit

337) Question 1

```
//Assume all the required imports are added
```

```
public class TestDemo{
```

```
String a1[]{"one","Two","three"};
```

```
String a2[]{"one","Two","three"};
```

```
@Test
```

```
public void test()
```

```
{
```

```
//Line 1
```

```
}
```

```
}
```

Choose the wrong option?

If we place Assert.assertEquals(a1, a2); at Line 1 the Test case will pass as it verifies the contents.

If we place Assert.assertSame(al, a2); at Line 1 the Test case will fail as it will check whether they point to same object

If we Place Assert.assertSame (al, a2); at Line 1 the Test case will pass as it verifies the contents

If we Place Assert.assertSame (al, a2); at Line 1 the Test case will pass as it verifies the contents

If we place Assert.assertEquals(al, a2); at Line 1 the Test case will pass as it verifies the contents

338) Question 2

Predict the output for the below code:

```
//Assume all the required import statements are added
```

```
public class TestDemo {  
    @Before  
    public void beforeTest10 {  
        System.out.println("in before test2");  
  
        @Before  
        public void beforeTest20 {  
            System.out.println("in before testi");  
  
        @Test  
        public void testO {  
            String a = "123";  
            Assert.assertEquals("123", a);  
        }  
}
```

- a. Test Passes as they point to the same object and prints the below in console

in before test1

in before test2

b. Test Fails and prints the below in console

in before test2

in before test1

c. Test Passes as they point to the same object and prints the below in console

in before test2

in before test1

d. Test Fails and prints the below in console

in before test1

in before test2

Answer: Actually compilation will fail as Junit accept only one @Before Annotation

339) Question 3

Consider a JUnit Test Class with /Unit Fixture annotations and the methods as below Question 3

@BeforeClass init()

@After Class --- close()

@Before --- UP()

@After --- tearDowno

@Test testSum10

@Test --- testEven10

In which order @Catethe methods will execute?

a. inti() setUp() testSum1() tearDown() setUp() testEven1() tearDown() close()

b. init() close() setUp() testEven1() setUp() testSum1() tearDown() tearDown()

c. init() setUp() testSum1() close() tearDown() setUp() testEven1() tearDown()

d. init() testSum1() setUp() tearDown() setUp() testEven1() tearDown() close()

Question set -2

340) Consider the following code snippet:

```
interface Bank
default void setBankName(String name)
this.name = name: (not declared)
default String getBankName()
return this.name;
class Customer implements Bank
protected String name;
public class UserInterface {
public static void main(String[] args)
Customer customer = new Customer();
customer.setBankName("MoneyBank");
System.out.println(customer.getBankName());
```

What will be the output when the above code gets executed?

- The code will not compile as name cannot be resolved in setBankName() of Bank interface
 - MoneyBank
 - The code will not compile due to compilation error in Customer class as name cannot be declared inside it
 - The code compiles but the output will be null.
-

341) Question 1

What will be the output of the code given below?

```
public static void main(String[] args)
LocalDateTime datel = LocalDateTime.now();
System.out.println(datel.isAfter(datel.plusDays(1)));
```

- False
- Two
- Error as we cannot call plusDays() method inside another method
- Nothing will get printed as isAfter() method has void as return type

342) Question 5

Which of this statements 0 not correct and will lead to compilation error? (Assume all the import statements are given properly)

- HashSet hs = new LinkedHashSet
- HashSet set = new HashSet 0:
- List list = new Vector 0;

(all compile, but Set should be used for HashSet = as best practice)

343) Question 1

Which of the following approach can be used to check whether a List is empty or not? Assume the List declaration is given as : List<String> employeeList = new ArrayList<>0:

- employeeList.size() == 0
- employeeList.isEmpty(null) can use .isEmpty();
- employeeList.equals(null), arrayList is defined, simply empty, and even in the case of null = null, that is not true
- employeeList.contains(null), for it to contain null, you would need to add it

344) Given:

```
public class TestDerno {  
    public static void main($1rlngn args){  
  
        Class myclass.Class.forNameTestDemol://line 1  
    }  
}
```

In the above code snippet Line 1 gives a compilation error that says 'Unhandled Exception Type ClassNotFoundException.. What IS the reason behnd this compilation error?

- ClassNotFoundException is a checked exception
- ClassNotFoundException is an unchecked exception
- ClassNotFoundException as Class is not defined
- ClassNotFoundException as myClass is not created using new keyword.

345) What will be the definition of display method in ParentClass which is also overnden in ChildClass ?

```
Public Class ParentClass {  
    //line1{  
    String signature = "(String, Integer[])";  
    System.out.println( str + " +signature);  
    return 1;  
}
```

```

}

public class ChildClass extends ParentClass
public int display(String str, Integer... data) throws Exception {
String signature = "(String, Integer())";
System.out.println(Overridden: " + str + " + signature);
return 0;
}
public static void main(String... args)
ChildClass cc = new ChildClass();
try {
cc.display("hello". 3);
catch (Exception e)
}}}

```

- public int display(Integer... data, String str) throws Exception
- public int display(String str, Integer data) throws Throwable
- public int display(String str, Integer... data) throws Exception
- public int display(String str, Integer... data)

346) Which of the following keyword is used to propagate an exception to its calling environment?

- raise
- throws
- catch
- thrown

347) What is the result when the following code snipped is compiled?

```

class Student extends Exception {

}

class Hosteller extends Student {

}

public class StudentTester {

public static void main(String args[])

try {

// Some monitored code

```

```

throw new Hosteller();

catch(Student st {
System.out.println("Student class exception");
}

catch(Hosteller host) {

System.out.println("Hosteller class exception");

}
}

}

```

- The Code will compile successfully and prints Student class exception
- The Code will compile successfully and prints Hosteller class exception
- The code will not compile Unreachable catch for Hosteller because Student class exception is caught before Hosteller class
- The code will not compile because Hosteller is not throwable

Since Hosteller is a more specific exception than Student, it should be caught before, otherwise Student will catch it, and “catch Hosteller” will never be reached.

348) Which of the following data structure is used by Varargs in Java?

- LinkedList
- Array
- ArrayList
- Vector

349) Which of the following is the correct syntax to declare the abstract method 'evaluate' with a Varargs variable 'marks'?

- public abstract double evaluate(double... marks, int rollNo);
- public abstract void evaluate(Integer rollNo, Float... marks);
- public abstract Float evaluate(Integer rollNo, Double..., marks);
- Varargs can't be used as a formal parameter in abstract

Varargs is last parameter, and signature is “type...name”

350) Question 1

If the child class of an abstract class does not implement all its abstract methods then it should be declared as?

- Abstract class
- Non Abstract class

- Static class
- Interface

351) Question

```
Interface Component {  
    String cname = Motor;  
    String getName(String name);  
}  
  
public class Demo implements Component {  
    public String getName(String name) {  
        System.out.println("Inside Demo class ");  
        return "Component from interface is : " + cname + " and component from class is. " + name + ", ";  
    }  
  
    public static void main(String[] args)  
    {  
        Demo demo = new Demo();  
        System.out.println(demo.getName("Battery") + " ");  
        demo.getName("Battery");  
    }  
}
```

- Inside Demo class. Component from interface is Motor and component from class is :Battery.
- Inside Demo class. Component from interface is Motor and component from class is :Battery.
- OInside Demo class.
- Inside Demo class. Component from interface is Battery and component from class is Battery.

352) Public class innerclassdemo

```
Private int bookid = 110;  
  
class Book {  
    private int bookId = 231;  
    private int getBookId()  
    {  
        return bookid;  
    }  
    private int getBookId()  
}
```

```

{
    return bookid;
}

public static void main(String[] args)
InnerclassDemo innerclassDemo= new InnerclassDemo:
/line1

System.out.println(innerclassDemo.getBookId()+"'" + book.getBookId());

```

Which of the below code fragment can be inserted at line 1 helps to get the output as 110231

- InnerClassDemo book = innerClassOemanew Book()
- **InnerClassDemo.Book book = innerClassDerno.new Book();**
- InnerClassDemo book = new Booka
- Book innernassDemo.book = 'noel ClassDemo.new Book0;

353) Question 3

Which of the following statement is correct about Singleton design pattern?

1.This type of design pattern comes under creational pattern

2.This pattern involves a single class which is responsible to create an object which make sure that only single object gets created

3 singleton class provides a way to access the object directly without the need to instantiate the object of the class

- only statement 1
- only statement 2
- only statement 3
- **All the statements are true**

354) Q2:

which of the following line will throw a compilation error and why?

```
class Apple { //line 1
```

```
int quantity; //line 2
```

```
class Main public static void main(String args[]) { //line 3
```

```
Apple apple; //line 4
```

```
System.out.println(apple.quantity);
```

- ```

 }
}

 - line 1, Because a class must be declared as public.
 - line 2, Because quantity must be initialized.
 - line 3, Because main method must be defined inside a public class.
 - line 4. Because apple has not been initialized
```

### Question set -3

355) 1) Consider the below code snippet:

```

interface Student {
 int student_id=101;

}

class Studentimpl implements Student{
 void getStudentId() {
 student id=102;
 }
}

```

What will happen when the above code compiles?

- a) Compilation Error as student\_id is static field and hence we cant change its value after initialization.
- b) The code will compile successfully.
- c) The code will compile successfully but when executed , it will lead to runtime exception as student\_id field cannot be changed.
- d) Compilation error as student\_id will not be visible in StudentImpl

356) 2) Consider the below code snippet:

```

public class TestDemo {

 public static void main(String[] args) {
 // TODO Auto-generated method stub
 int i=4;
 int j=4;
 System.out.println(i==j);
 Integer k=new Integer(100);
 Integer l=new Integer(4);
 System.out.println(k==l);
 }
}

```

357) What would be the issue in the above code?

- a) Comparing Wrapper classes objects using == is wrong as it checks if both points to same object
- b) Comparing primitive data types using == is wrong
- c) Both A and B
- d) No issues in the above code

358) 3) Which of the below statement indicate the need to use of Factory pattern?

- a) when we want two classes to use the same object
- b) we only want one instance of the object to exist
- c) we want to build a chain of objects
- d) when we want to allows the sub-classes to choose the type of objects to create

359) 4) which of the following statements are true if a duplicate element objT is added to a HashSet?

- a) The element objT is not added and add() method returns false.
- b) the element objT is added successfully
- c) An exception occurs during runtime.
- d) An exception occurs during compile time.

360) 5) Which among the following is/are true about the design pattern? Select any two option

- a) Design pattern depends upon abstraction
- b) Design patterns are completed designs that can be transformed directly into code.
- c) Design pattern depends on abstraction, follows the process of dependency injection.
- d) Design pattern is a template of solving problem that can be used in many real world software

361) 7) Refer the below code snippet and predict the outcome

```
public class RepeatingAnnotations{
 @Retention(RetentionPolicy.RUNTIME)
 public @interface Chocolates{
 Favourite[] value();
 default{};
 }
}
```

```

}

@Repeatable(value = Chocolate.class)

public @interface Favourite{
 String value();
}

@Favourite("Dairy Milk")
@Favourite("Kitkat")
@Favourite("5 Star")
@Favourite("Galaxy")

public interface Chocolate{
}

public static void main(String args[]){
 Chocolates Chocolate = Chocolate.class.getAnnotation(Chocolates.class);
 for(Favourite favourite:chocolates.value()){
 System.out.println(favourite.value());
 }
}

```

**No Answer**

362) 8) Which of the below code is implemented without best practice standard

i) String[] str = new String[]{"Hi", "Hello", "Welcome"}

```
List strList = Arrays.asList(str)
```

```
For(Iterator itr = strList.iterator();itr.hasNext();){
```

```
System.out.println(itr.next());}
```

ii) Integer i1 = new Integer(11);

```
Integer i2 = new Integer(11);
```

```
System.out.println(i1==i2)
```

a) Option(i) doesn't follow best practices can be improved using for (String S: strList)

b) Option(ii) doesn't follow best practices as objects should not be compared with ==

c) **Option(i) and (ii) are not following best practices.**

d) Option(i) and (ii) are following best practices.

363) 9) Which of the below are not good practice for creating objects?

- a) Lazy initialization of objects
- b) Creating String literals instead of String objects**
- c).Creating Wrapper objects instead of primitives
- d).Invoking static factory methods for immutable classes

364) 10) Which of the below method is valid as per java naming convention?

- a) METHOD\_NAME
- b) MethodName
- c) methodName**
- d) Method\_Name

365) 14) Given

```
public class AppTest {
 String message ="Hello";
 int length = message.length();
 @Test
 public void testOne(){
 System.out.print(length + " ");
 assertEquals(length,5);
 }
 @Test
 public void testTwo(){
 System.out.print(length + " ");
 assertEquals(length,5);
 }
 @After
 public void teardown() {
 length = length +1;
 }
}
```

```
System.out.print(length + " ");}}
}}
```

What is the result?

Answer : Both test will pass and print 5 6 6 7 in the console

366) 15) Which of the following keyword is used to prevent the content of a variable from being modified from outside?

- a) final
- b) constant
- c) static
- d) transient

367) 16) What is the output for the below code?

```
//Myexception is custom exception class
public class TestDemo{
 static void myCode() throws MyException {
 try{
 throw new MyException("TestException");
 } catch (Error | Exception ex) {
 System.out.print("Inside Error and Exception")};
 }
 public static void main(String[] args) throws MyException {
 try {
 myCode();
 } catch(Exception ex){
 System.out.print("Inside Exception");}}}

d) prints "Inside Error and Exception"
e) An exception is thrown at runtime
f) Prints "inside Exception"
```

368) 17) Which of the following are true about enums in java?(Choose any 3)

- a) Enums can implement any other interface in java
- b) An instance of enum can be created outside of enum itself
- c) Enums cant extend any other class except the abstract base class java.lang.enum
- d) Enums can be used to implement Singleton design pattern

369) 18) Predict the output of the following code?

//Assume all the required imports are added

```
public class TestDemo{
```

```
@Test
```

```
public void test() {
```

```
String a = “ ”;
```

```
Assert.assertNotNull(a);}}
```

- e) Test Passes
- f) Test fails
- g) An exception is thrown at runtime
- h) Compilation fails

370) 19) Which of the following statement regarding an abstract class are true in java?

i) Object of an abstract class cant be created

ii) An abstract class is designed only to act as a base class in hierarchy to be inherited by other classes.

- a) Only i
- b) Only ii
- c) Both I and ii
- d) Neither I and ii

**Not sure which one is correct**

371) 20) what is the output of the following code?

```
package exceptions;
```

```
public class ExceptionDemo {
```

```
static class Car implements AutoCloseable {
```

```
public void close() {
```

```
System.out.print(“Automatic Door close”)}}

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

```

try{
 Car car = new Car();
 System.out.print("inside try block");
}

Catch(Exception e){
 System.out.print("Inside catch block")
}

finally{
 System.out.print("finally")
}
}

```

**Answer : Automatic Door Close inside try block finally**

372) 23) Given

```

public class Util {
 public static void main(String[] args){
 String name = "Martin";
 Name.concat="Luther";
 System.out.println(name.concat("King"));
 }
}

```

Predict the output:

- e) Martin
- f) Martin Luther King
- g) Martin Luther
- h) Martin King**

373) 24)Analyze the below code and predict the outcome when compiled and executed?

```

public class Demo extends Book {
 int bookid=4567;
}

```

```

public int getValue() {
 return bookid;
}

public void call() {
 System.out.print(super.getValue()); //Line 1
}

Public static void main(String[] args) {
 Book book = new Book();
 Super.call(); //Line2
}

class Book {
 int bookid =17897;
 public int getValue() {
 return bookid;}}

```

**Answer : 17897**

374) 25) What is the output of the following code?

```

class Employee {
 void display(char c) {
 System.out.println("Employee name starts with : "+c+");

 System.out.println("His Experience is 11 years")};

 class Main extends Employee {
 void display(char c) {
 super.display(c);

 System.out.println("Another employee name also starts with: "+c+");

 new Employee().display("D");

 display(7);
 }
 }
}

```

```
String display(int c){
 System.out.println("His experience is : +c" years);
 return "Bye";}
No Answer
```

375) 26) Given

```
public class Sample {
 public static void main(String[] args) throws Exception {
 try {
 System.out.println("In try block");
 System.exit(0);
 } catch(Exception ex){
 System.out.println("In catch block");
 ex.printStackTrace();
 } finally{
 System.out.println("In finally block");
 }
 }
}
```

Predict the Output

- c) In try block In finally block
- d) In try block

376) 27) Given

28) What is the result when the following code snippet is compiled?

```
class Tester {
 public static void main(String[] args) throws Exception {
 try {
 throw new IOException(); //Line1
 } catch(IOException | Exception e) { //Line2
 System.out.println(e + "caught"); //Line3
 }
 }
}
```

}}}

- e) The code will compile and prints IOException caught
- f) The code will not compile due to error in Line 2 i.e. IOException already caught.
- g) The code will not compile and shows compile time error in Line 2 and Line 3
- h) The code will compile and prints java.IOException caught

377) 29) What changes need to be made in the following code to make the singleton pattern correct?(Choose any 2)

```
public class Employee {
 public static Employee employeeInstance;
 private Employee() {}
 public static Employee getEmployee()
 {
 if(employeeInstance==null){
 employeeInstance = new Employee();
 }
 return employeeInstance;}}
```

- e) None of the Singleton Pattern is properly implemented
- f) Rename employee to Instance
- g) Add synchronized to getEmployee()
- h) Change the access modifier of employeeInstance from public to private

No Answer

378) 30) which of the following interface should be implemented by a resource, if we want to create the object of that resource in the resource part of a try block?

- a) Closeable
- b) Closeable<E>
- c) AutoCloseable
- d) AutoCloseable<E>

379) 32) what is wrong with respect to the code that is given below

```

class Student {
 protected static String mailDomain = "@infosys.com";
 //instance variables and methods follow
}

class Hostelite extends Student{
 public void generatedReport(String name)
 {
 System.out.println(name+Student.mailDomain);}
}

```

- e) The code will not compile as the static variables of a class cannot be used inside the instance specific methods of the child class.
- f) The Code will compile but will not get executed as the static variables of a class cannot be used inside the instance specific methods of the child class
- g) The code seems to be perfect and will work as expected**
- h) The code will not compile as the keyword, implements is not used for inheriting from the parent class

380) 33) public class TestDemo {

```

public void main(int x) {
 System.out.println("Main1")
}

public static void main(String args[]){
 System.out.println("Hello Main");}

```

- a) Main1
- Hello Main
- b)Hello Main
- Main1
- c) Main1
- d)Hello Main**

381) 35) which of the following mechanism in OOP is a process of hiding certain details and showing only essential information to the user?

- a) Polymorphism

b) Encapsulation

c) Abstraction

d) Inheritance

382) 38) Given

```
Class ArrayDemo{
```

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

```
 int x[] = display();
```

```
 for (int i=0; i<x.length; i++)
```

```
 System.out.print(x[i]+” “);
```

```
}
```

```
Public static int[] display(){
```

```
//Line1
```

```
}
```

```
}
```

383) 39) public class Project{

```
 Private Integer projectId;
```

```
 Private String projectName;
```

```
 Public static void main(String[] args){
```

```
 Project oldProject=null;
```

```
 Project newProject=null;
```

```
 oldProject=new Project();
```

```
 newProject=new Project();
```

```
 newProject=oldProject;
```

```
 oldProject=new Project();
```

```
}
```

```
}
```

Which of the following statement is true after the code gets executed?

- e) oldProject and newProject are referring to different objects.
- f) oldProject and newProject are referring to same objects.
- g) oldProject is referring to two different objects at the same time.
- h) newProject is referring to three different objects at the same time.

384) 41) which of the following statements are FALSE?

(Choose 2 options)

- a) An Interface can extend from only one interface.
- b) A class can extend from another class and at the same time implement any number of interfaces.
- c) A class can extend multiple abstract classes.
- d) Many classes can implement the same Interface.

385) 42) what is the result when the following code is compiled and executed?

```
Class Demo{
```

```
int x = 1;
int y = 2;
```

```
Demo display(Demo demoParam)
```

```
{
 Demo obj=new Demo();
 obj=demoParam;
 obj.x=demoParam.x++ + ++demoParam.y;
 demoParam.y=demoParam.y;
 return obj;
}
```

```
Public static void main(String[] args){
```

```
Demo obj1=new Demo();
Demo obj2=obj1.display(obj1);
```

```
System.out.println("obj1.x = " + obj1.x + "obj1.y = " +obj1.y);
System.out.println("obj2.x = " + obj2.x + "obj1.y = " +obj2.y);
}
}
```

- e) obj1.x=4 obj1.y=4  
    obj2.x=4 obj1.y=4
- f) obj1.x=3 obj1.y=3  
    obj2.x=4 obj1.y=3
- g) obj1.x=4 obj1.y=3  
    obj2.x=4 obj1.y=3
- h) obj1.x=3 obj1.y=4  
    obj2.x=4 obj1.y=3

Answer: No Answer

386) 43) What is wrong with respect to the code that is given below.

```
Class Student
{
 Protected static String mailDomain = "@infosys.com";
 //instance variables and methods follow

}

Class Hostelite extends Student
{
 Public void generateReport(String name)
 {
 System.out.println(name+Student.mailDomain);
 }
}
```

- a) The code will not compile as the static variables of a class cannot be used inside the instance specific methods of the child class
- b) The code will compile but will not get executed as the static variables of a class cannot be used inside the instance specific methods of the child class
- c) The code seems to be perfect and will work as expected
- d) The code will not compile as the keyword implements is not used for inheriting from the parent class.

387) 44) Predict the output of the following code:

```
Public Class Main{
 Public void display(int i)
 {
 System.out.println("inside first");
 }
 Public void method(int i, int j)
 {
 System.out.println("inside second");
 }
 Public void method(int...k)
 {
 System.out.println("inside third");
 }
 Public static void main(String[] args){
 {
 new Main().method(110);
 new Main().method(110,210);
 new Main().method(110,210,310); //Line1
 }
}
```

```
new Main().method(110,210,310,410); //Line2
}}
```

- e) inside first  
    inside second  
    inside third

- f) inside first  
    inside second  
    inside third  
    inside third

- g) inside third  
    inside second  
    inside third

- h) inside third  
    inside second  
    inside third  
    inside third

388) 45) What is the result when the following code is compiled and executed?

```
Public class Test{
 Public void method(){
 for(i = 0; i<3; i++){
 System.out.print(i);
 }
 }
 Public static void main(String[] args){
 method();
 }
}
```

- e) 012
- f) 0 1 2 3
- g) Compilation fails as cannot make a static reference to the non static method.
- h) 2 1 0

389) 46. Which datatypes can be stored by Java collections?

- A. Boolean
- B. int
- C. Byte
- D. Character

390) 47.Which among the following option are correct with respect to HashMap?

- A. override boolean equals(Object o)
- B. override toString()
- C. override hashCode()
- D. override String hashcode()

391) 48. How many numbers of values can be accommodated by Varargs in Java?

- A. Atleast one values
- B. Atmost two values
- C. More than one value
- D. Any number of values

392) 49. Which of the following keyword can be used to restrict a class to be inherited in Java?

- A. Abstract
- B. final
- C. constant
- D. private

393) 52) Which of the following is a necessary condition for implicit type conversion in Java?

Option A: The destination type is smaller than source type

Option B: The destination type is larger than source type

Option C: The destination type can be larger or smaller than source type

Option D: The destination and the source type are the same.

394) 53) Predict the output of the below code snippet?

```
Collection sorted = new LinkedList();
```

```
Sorted.add("B");
```

```
Sorted.add("C");
```

```
Sorted.add("A");
```

```
For(Object object : sorted){
```

```
 System.out.print(object + ",");
```

```
}
```

Option A: A,B,C

Option B: B,C,A

395) 54) Which of the following Jump statement can skip processing of one iteration if a specified condition occurs and remaining iterations?

Option A: break

Option B: return

**Option C: continue**

Option D: exit

396) 55) Which of the following is false regarding parameterized constructors in Java?

**Option A: Parameterised constructors should have void as return type**

Option B: Parameterised constructors can take any number of parameters

Option C: Parametersied constructors cannot have private access modifier

Option D: Parameterised constructors cannot throw an exception

397) 56) Ria has a class called 'Account.java' under tow separate packages com.infy.debit and com.infy.credit. Can she use the Account class of both the packages in another class called 'ReportUti.java' of package com.infy.util?

Option A; Yes, she can use

**Option B: No, she cannot as there will be a compilation eror stating the import collides with another import**

Option C: No, she cannot. The code will pass compilation but an ambiguity will get encountered during the execution.

Option D: No, she cannot as there will be a compilation error whiel creating Account class for the second time through in a different

398) 57) What is the result of the following code?

```
Public class Vehicle{
```

```
 Static class Car{
```

```
 Public void go(){
```

```
 System.out.println("Car Ignition");
```

```
}
```

```

}

Static class ElectricCar extends Car{
 Public void go(){
 System.out.println("ElectricCar Ignition");
 }
}

Static class PetrolCar extends Car{
 Public void go(){
 System.out.println("PetrolCar Ignition");
 }
}

Public static void main(String[] args){
 Car car = new ElectricCar();
 Car.go();
}

<<<No Options>>>

```

399) 58) Predict the output of the below code snippet?

```

ArrayList list = new ArrayList();
List.add("Infosys");
List.add("Google");
For(String s:list){
 System.out.print(" "+s);
 List.clear();
}

```

Option A: It prints Infosys

Option B: Compilation fails as the line "for(String s:list)" cannot convert from elementtype

400) 59) What is the output for the below code snippet?

```

Public class TestDemo{
 Public static void main(String[] args){
 Try{
 Int a = 20/20;
 Int b = 20/10;
 System.out.println("a="+a+"b="+b);
 Try{
 If(a==1){
 A=a/(a-a);
 }
 If(b==2){
 Int c[]={1};
 C[22]=99;
 }
 }
 Catch(ArithmeticException ae){
 System.out.println("ArithmeticException Bock 1");
 }catch(ArrayOutOfBoundsException ai){
 System.out.println("ArrayIndexOutOfBoundsException Block");
 }
 Catch(ArithmeticException ae){
 System.out.println("ArithmeticException Block 2");
 }
 }
 <<<No Options>>>
 }
}

```

401) 60) Given:

```

Ex.printStackTrace();
}finally{
 System.out.println("In finally block");
}

```

```
}
```

```
}
```

```
}
```

Predict the output?

Option A: In the try block in finally block

Option B: In try block

Option C: In try block in Catch block in finally block

Option D: The code will not compile due to Sysntax.exit(1);

<<<No Ans>>>

402) 61) If the Child class of an abstract class does not implement all its abstract methods then it should be declared as?

Option A: Abstract class

Option B: Non Abstract class

Option C: Static class

Option D: Interface

403) 62) Which of the following pattern refers to creating duplicate object while keeping performance in mind?

Option A: Builder Pattern

Option B: Bridge Pattern

Option C: Prototype Pattern

Option D: Filter Pattern

404) 63) Given

```
Class Aircraft{
```

```
String name = "MiG";
```

```
String start(){
```

```
Return "main engine Start";
```

```

}

}

Class CivilianAircraft extends Aircraft{

String name = super.name;

String start(){ //Line 1

Return "Secondary engine start";

}

}

Public class Demo{

Public static void main(String[] args){

New Demo().go();

}

Void go(){

Aircraft aircraft = new CivilianAircraft(); //Line2

System.out.println(aircraft.name + " " + aircraft.start());

}

}

```

Option A: MiG Main engine start

**Option B: MiG Secondary engine start**

405) 64) What is the result when the following code is compiled and executed?

```

Public class Test{

Public void method(){

For(int i=0;i<3;i++){

System.out.print(i);

}

}

Public static void main(String args[]){

Method();

```

}

**Option A: 012**

Option B: 0 1 2 3

Option C: Compilation fails as cannot make a static reference to the non static method

Option D: 2 1 0

406) 65) class Hostelite extens Student

{

    Public void generateReport(String name)

    {

        System.out.println(name+Student.mailDomain);

    }

}

Option A: The code will not compile as the static variables of a class cannot be used inside the instance specific methods of the child class

Option B: The code will compile but will not get executed as the static variables of a class cannot be used inside the instance specific methods of the child class

**Option C: The code seems to be perfect and will work as expected**

Option D: The code will not compile as the keyword, implements is not used for inheriting from the parent class

407) 66) What is the output of the following code?

Class Employee{

    Void display(char c){

        System.out.println("Employee name starts with: "+c".");

        System.out.println("His experience is: 11 years.");

    }

}

Class Main extends Employee{

    Void display(char c){

        Super.display(c);

```
 System.out.println("Another employee name also starts with:"+c+ years.");

 New Employee().display(D);

 Display(7);

 }

String display(int c){

 System.out.println("His experience is:"+c+"years.");

 Return "Bye";

}

}
```

```
Public class Demo{

Public static void main(String a[]){

}
```

Option A:

His experience is: 7 Years

His experience is: 11 years

Option B:

Employee name starts with: S

His experience is: 11 years

Another employee name also starts with: S

His experience is: 7 years

Option C:

Employee name starts with: S

His experience is: 11 years

Another employee name also starts with: S Employee

His experience is: 11 years

His experience is: 7 years

Option D:

Employee name starts with: S

His experience is: 11 years

Another employee name also starts with: S

Employee name starts with: D

<<<No Ans>>>

408) 67) Predict the output of the below code

Class Car{

```
Void start(){
 System.out.println("car Starts");
}
}
```

Class Bike{

```
Void start(){
 System.out.println("Bike Starts");
}
}
```

Class Automobile extends Car{

```
Void start(){
 System.out.println("Automobile Starts");
}
}
```

Public class ExceptionDemo{

```
Public static void main(String[] args){
 System.out.println("implementing type casting");
 Car d = new Car();
 Automobile automobile = (Automobile)d;
}
```

<<<No Options>>>

409) 68) Predict the output of the following code:

```
Public class Main{
 Public void display(int i)
 {
 system.out.println("inside first");
 }
 Public void method(int l, int j)
 {
 System.out.println("inside second");
 }
 Public void method(Int i...k)
 {
 System.out.println("inside thord");
 }
 Public static void main(String args[])
 {
 New main().method(110);
 New main().method(110,210);
 New main().method(110,210,310); //Line 1
 New main().method(110,210,310,410); Line 2
 }}
a) inside first
 inside second
 inside third
 inside third
```

69

410) 70) public class TestDemo{

```
@Before
Public void beforeTest1(){
 System.out.println("in before test2");
}

@Before
Public void beforeTest2(){
 System.out.println("in before test 1");
}

@Test
Public void test(){
 String a = "123";
 Assert.assertSame("123,a);
}
}
```

<<<No Options>>>

Ans: C

- 411) 71. Which of the following Java component can't be referred using 'super' keyword?
- A.public constructor
  - B.protected method
  - C.private method
  - D.protected instance variable
- 412) 72. Which of the following are the advantage of exception handling in Java(choose any 3 option)?
- A.To maintain the normal flow of execution
  - B.Meaningful error reporting
  - C.To document compile time errors
  - D.To prevent the abrupt termination of a program

Ans : A,B,D

- 413) 73. What is the correct way of placing "this" keyword in a constructor?

- A.First statement
- B.Last Statement
- C.Inside a parameterized constructor only
- D.Can't be placed at any line in constructor

**Ans : A**

414) 74. At which position should Varargs be placed in a parameterized method?

- A.First place
- B.Last Place
- C.Second Last place
- D.Can be anywhere

**Ans : B**

415) 75. Which of the following is a valid lambda expression?

- A.(sum) → true
- B.x,y → true
- C.sum → {return 1 ==1}
- D.(a,b) → {int result; return result>0;}

**No Answer**

416) 76. //Assume all the required imports are added

```
public class TestDemo{
 String a1[] = { "one", "Two", "three" };
 String a2[] = { "one", "Two", "three" };

 @Test
 public void test(){
 // Line 1
 }
}
```

Choose the wrong option?

**Answer: If we place Assert.assertEquals(a1,a2): at Line1 the test case will pass as it verifies the contents**

417) 77. What will be written at Line 1 so that the below code will compile and run successfully?

```
Public class Main {
 // line 1
 static {
 X[0]=102;
 } public static void main(String[] args) {
 System.out.println(x[0]);
 }
}
```

Ans : static int[] x = new int[3]

```
418) public class D1{
void main(){
 System.out.println("JAVA");
}
static void main(String args){
 System.out.println("Spring");
}
public static void main(String[] args){
 System.out.println("Hibernate");
}
void main(Object[] args){
 System.out.println("Apache Camel");
}
```

}

What is the output?

- A. Hibemate
- B. Spring
- C. JAVA
- D. Apache Camel

2)

```
419) class Employee{
 public final void show(){
 System.out.println("show() inside Employee");
 }
}
final class Unit extends Employee {
 public void show1() {
 final int x=100;
 System.out.println("show() inside Unit");
 System.out.println(x);
 }
}
public class D2 {
 public static void main(String[] args) {
 Employee employee = new Unit();
 new Unit().show1();
 }
}
```

What will be the output when the above code is complied and executed?

- A. 100
- Show() inside Unit
- B. Show() inside Employee
- C. Show() inside Unit  
Show() inside Unit
- 100
- D. Show() inside Unit**
- 100**

**3)**

**420) What is the result of the following code?**

```
public class Branch {
 static class Customer {
 public void go() {
 System.out.println("Inside Customer");
 }
 }
 static class Account extends Customer {
 public void go() {
 System.out.println("Inside Account");
 }
 }
 static class Branch extends Customer {
 @Override public void go() {
 System.out.println("Inside Branch");
 }
 }
 public static void main(String[] args) {
 //Line 1
 }
}
```

**421) What will be the output when we add the below code at Line1 and execute the program?**

```
Customer customer = new Account();
Branch branch = (Branch) customer;
branch.go();
```

- A. Inside Customer
- B. Inside Account
- C. Inside Branch
- D. The Code does not compile because (Branch)Customer is incorrect
- E. An exception is thrown at runtime because (Branch)Customer is incorrect

**422) 4) Predict the output of the following code.**

```

public class D4 {
 public static void main(String[] args) {
 displayRegistration("Hockey"); //Line 1
 displayRegistration("Kho-Kho", 132, 102, 36); //Line 2
 }

 public static void displayRegistration (String gameName, int... id) {
 System.out.println("Registration for " + gameName + ".");
 for(int i=0; i<id.length; i++)
 System.out.println(id[i] + " ");
 }
}

```

A. Registration for Hockey:

Hockey  
 Registration for Kho-Kho:  
 Kho-Kho  
 132 102 36

B. Registration for Hockey:

Registration for Kho-Kho:  
 132 102 36

C. Registration for Hockey:

D. Registration for Hockey:

Hockey

5)

423) Public interface InterfaceDemo {  
 //Line 1  
 }

Select the suitable code fragment can be inserted at Line1. (Choose at that apply.)

(Checkbox)

A. Void display (int x);

```

B. Void display (int x){

}

C. Public static void display(int x){

}

D. default void display(int x){

}

E. public interface Demo {

}

```

\*\*\*\*\*

**424) 6) What is the output of the following code?**

```

class Employee {
 void disp(char c)
 {
 System.out.println("Employee name starts with : "+c+",");
 System.out.println("His experience is 11 years ");
 }
}

class Main extends Employee
{
 void disp(char c)
 {
 super.disp(c);
 System.out.println("Another employee name also starts with "+c+",");
 new Employee().disp('D');
 disp(7);
 }

 String disp(int c)
 {
 System.out.println("His experience is "+c+" years");
 return "Bye";
 }
}

public class Example13
{
 public static void main(String[] args)
 {
 Employee emp = new Main();
 emp.disp('S');
 }
}

```

}

- a) Employee name starts with : S. His experience is : 11 years. Another employee name also starts with : S. Employee name starts with : D. His experience is also
  - b) Employee name starts with : S. His experience is : 11 years. Another employee name also starts with : S. His experience is 7 years
  - c) Employee name starts with : S. His experience is : 11 years. Another employee name also starts with : S. Employee name starts with : D. His experience is als
  - d) Employee name starts with : S. His experience is : 11 years. Another employee name also starts with : S.
- 

**425) 7) Predict the output of the below code:**

```
class Dog {
 void show() {
 System.out.print("Dog");
 }
}

class Cat {
 void show() {
 System.out.print("Cat");
 }
}

class BullDog extends Dog {
 void show() {
 System.out.print("BullDog");
 }
}

public class D5 {
 public static void main(String[] args) {
 System.out.print("Implementing type Casting");
 Dog d = new Dog();
 BullDog bd = (BullDog) d;
 bd.show();
 }
}
```

- a) Display “Implementing type Casting” in Console.
- b) Display “Implementing type Casting” and “BullDog” in Console.
- c) RUNTIME ERROR : java.lang. ClassCastException

- d) Display “BullDog” in console.
- 

**426) 8) Given:**

```
public class D6 {
 public static void main(String[] args) {
 try {
 method();
 System.out.print("Inside try");
 } catch (RuntimeException ex) {
 System.out.print("Inside catch(RuntimeException)");
 } catch (Exception ex1) {
 System.out.print("Inside catch(Exception)");
 } finally {
 System.out.print("finally");
 }
 System.out.print("end");
 }

 public static void method() {
 // Line 26
 }
}
```

Which code fragment can be inserted at Line 26 to display the output as “Inside catch(RuntimeException) finally end” ?

- A. **throw new RuntimeException();**
  - B. **throw new Exception();**
  - C. **throws new RuntimeException();**
  - D. **throws new Exception();**
  - E. **throw new Error();**
- 

**427) 9) Given:**

```
public class ExceptionInClass
{
 int data=10;
 void calculate() throws Exception
```

```
{
 try
 {
 data++;
 try
 {
 data++;
 // Line12
 }

 Catch(Exception ex)
 {
 data++;
 }
 Catch(Exception ex)
 {
 data++;
 }
 }
}
Void display()
{
 System.out.println(data);
}
Public static void main(String[] args)
{
 ExceptionInClass exceptionInClass = new ExceptionInClass();
 exceptionInClass.calculate();
 exceptionInClass.display();
}
```

```
 }
}
```

428) Which of the below code fragment needs to be inserted at Line12 to display the output as 15.

A. try{

```
data++;
throw new Exception();
}Catch(Exception ex){
data++;
throw new Exception();
}
```

B. try{

```
data++;
throw new Exception();
}Catch(Exception ex){
}
```

C. try{

```
throw new RunTimeException();
}Catch(Exception ex){
data++;
throw new RunTimeException();
}
```

D. try{

```
throw new Exception();
}Catch(Exception ex){
data--;
throw new Exception();
}
```

**429) What will be the output of below code snipped ?**

//assume all the required imports are added

// assume all the required classes exists

```
@RunWith(Suite class)
```

```
@suite suiteclasses{
```

```
Sample class,
```

```
TestClass class
```

```
)}
```

```
Private class junittestsuite { }
```

- A. Test passes
- B. Test fails
- C. An exception is thrown at runtime
- D. Compilation fails**

```
@RunWith(Suite class)
```

```
@Suiteclasses({C1 class , C2 class})
```

```
Public class TestDemo{
```

```
@beforeclass
```

```
Public static void setUpClass(){
```

```
System.out.println("Master setup");
```

```
}
```

```
@before
```

```
Public void setup(){
```

```
System.out.println("Slave setup");
```

```
}
```

```
@test
```

```
Public void test(){
 Assertnotsame(2,2)
}

@afterclass{

 Public static void teardownclass(){
 System.out.println("Master teardown");
 }

 @after

 Public static void teardown (){
 System.out.println("tear teardown");
 }
}
```

- a) 3 testcases runs and in console the output is displayed as

Master setup

Master teardown

- b) 2 testcases runs and in console the output is displayed as

**Master setup**

**Master teardown**

- c) 3 testcases runs and in console the output is displayed as

Master setup

Slave setup

Slave teardown

Master teardown

- d) 2 testcases runs and in console the output is displayed as

Slave setup

Slave teardown

Slave setup

Slave teardown

- e) Compilation error in testdemo class

**430) //Assume that the first two of three test cases fail in “Testclass”**

// Assme all the required import statements are added

```
Public class testrunner{
 Public static void main(String [] args){
 Result result = junitcore.runclasses(testclass.class)
 For (Failure failure : result.getfailures()){
 System.out.println(result.wassuccessful());
 }
 }
}
```

- A. False
- B. True
- C. False false true
- D. False false false

**431) Consider the below code snippet**

```
Locale locale = new Locale("da","DK");
NumberFormat numberFormat = NumberFormat.getInstance(Locale);
String number = numberformat.format(100.99);
System.out.println(number);
```

Here NumberFormat.getInstance() follows which design pattern ?

- a) Factory method pattern
- b) Singleton pattern

c) Abstract Factory Pattern

d) Builder pattern

432) **Which** of the below statement indicate the need to use factory pattern ?

- A. We have two classes that do the same thing
- B. We only want one instance of the object to exist
- C. We want to build a chain of objects

**D. We don't want the caller to depend on a specific implementation**

433) What changes need to be made in the following code to make the singleton pattern correct ?

```
Public class Employee{
```

```
 Public static Employee C;
```

```
 Private Employee(){}
}
```

```
 Public static Employee getEmployee()
```

```
{
```

```
 If(employeeInstance==null){
```

```
 employeeInstance=new Employee();
```

```
}
```

```
 Return employeeInstance;
```

```
}
```

```
}
```

**checkbox**

1. None of the singleton pattern is properly implemented

2. Rename employee to instance

3. Rename getEmployee() to getInstance()

4. Change the access modifier of employeeInstance from public to private

5. mark employee final

6. Add synchronized to getEmployee()

434) Which of the below code is implemented without best practices standard ?

```

1. List list;

public List getList{
 If(list.size()==0)
 Return null;
 Else
 Return list;
}

2. Integer i1=new Integer(11);

Integer i2=new Integer(11);
System.out.println(i1==i2);

3. String[] str=new String[]{"Hi","Hello","Welcome"};
List strList=Arrays.asList(str);
For(iterator itr=strList.iterator();itr.hasNext();){
 System.out.println(itr.next());
}

1.Option(i) is valid
2.Option(ii) is valid
3.Option(ii) and (iii) are valid
4.option(i) and (ii) are valid

```

**435) Identify the valid code to be inserted at Line5, assume the code is running in multithreaded environment?**

```

Public class Demo{
 Private static String id;
 Private Random random =new Random();
 Public Demo(){
 //Line5
 Id="ACC1101"+Math.abs(random.nextInt());
 }
}

```

```

 }
 }

Public String getId(){

 Return id;

}

1.if(id==null){
2.synchronized(this){if(id==null){

3.if(id==null){synchronized(this){

4.synchronized{if(id==null){


```

436) Select the valid code fragment according to Java coding standard?

1. Public void draw(String s){
   
    If(s.equals("Square")){
   
        drawSquare();
   
    }
   
    If(s.equals("Rectangle")){
   
        drawRectangle ();
   
    }
 }
  2. Public void draw(String s){
   
    If("Square".equals(s)){
   
        drawSquare();
   
    }
   
    If("Rectangle".equals(s)){
   
        drawRectangle ();
   
    }
 }
- 1.only option (i) is valid**
2. only option (ii) is valid
  - 3.Both (i) and (ii) are valid
  4. Both (i) and (ii) are invalid

**437) Given**

```
Public class TestDemo{
 Private static Object staticObject;

 Public static Object createStaticObject(){
 If(staticObject==null){
 staticObject= new Object();
 }
 Return staticObject;
 }
}
```

What changes are required in the above code for successful execution?

- 1.The method createstaticObject should be synchronized
2. The method createstaticObject should be private
- 3.The staticObject reference should not be static
4. The method createstaticObject should not return Object type

**438) What will happen to the following code when trying to get compiled?**

```
Class RepeatableAnnotation
{
 @SuppressWarnings("all")//line 1
 @SuppressWarnings("deprecation")//line 2
 Public void over()
 {
 New Date().setDate(00);
 }
}
```

- 1.Unreachable code error will be generated at line 2

2.Compilation will not be successful as @SuppressWarnings annotation is non-repeatable in nature

3.warnig will be issued as it is totally unnecessary to mention @SuppressWarnings("deprecation")

4.code will get complied successfully with out any warning

**439) What is true regarding the following code snippet?**

Interface StaticInterface

```
{
Static void staticMethod()
{
System.out.println("inside interface");
}
}
```

Class StaticInterfaceImpl implements StaticInterface

```
{
Public void staticMethod()
{
System.out.println("inside class");
}
}
```

Public class StaticDemo

```
{
Public static void main(String[] args)
{
New StaticInterfaceImpl() staticMethos();
}
}
```

1.code will not get complied as the static method should always be public

2. code will not get complied as the static method is overridden in StaticInterfaceImpl

3.code will print "inside interface" on execution

4. code will print “inside class” on execution

**440) Refer the below code snippet and predict the outcome?**

```
Interface Interface1
{
 Default void method1()
 {
 System.out.println("Inside default method");
 }
}

Interface DefaultExtend extends Interface1
{
 Default void method1()
 {
 System.out.println("Default method redefined");
 }
}

Public class InterfaceWithDefaultMethod implements DefaultExtend
{
 Public static void main(String[] args)
 {
 InterfacewithDefaultMethod defaultExtend=new InterfaceWithDefaultMethod() //line4
 defaultExtend.method1(); //Line5
 }
}
```

1.Inside default method

2.Default methos redefined

3.Compilation fails at Line5

4.Runtime exception will be thrown at Line5

**441) Which of the following is incorrect regarding interfaces in Java SE\***

- a.all the methods are public,abstract by default
- b.all the variables are public,final by default
- c.methods can have implementation
- d.it's possible to hold static methods

1.a and b

2.b and c

3.a,b and c

**4.a only**

**442) What will happen when the following code is subjected to compilation and execution?**

```
Interface DefaultMethodInterface1{
 Default public void defaultMethod(){
 System.out.println("DefaultMethodInterface1");
 }
}

Interface DefaultMethodInterface2{
 Default public void defaultMethod(){
 System.out.println("DefaultMethodInterface2");
 }
}

Public class TestDemo implements DefaultMethodInterface1, DefaultMethodInterface2{
 Public static void main(String[] args){
 DefaultMethodInterface1 defMethIN=new TestDemo();
 defMethIN defaultMethod();
 }
}
```

}

- 1.An execption is thrown at runtime
- 2.Compilation fails**
- 3.DefaultMethodInterface1 will get printed on the console
4. DefaultMethodInterface2 will get printed on the console

**443) Given:**

```
Public class Demo11{
 Public static void main(String args[]){
 set numbers=new HashSet();
 numbers add(new Integer(45));
 numbers add(88);
 numbers add(new Integer(77));
 numbers add(null);
 numbers add(789L);
 Iterator iterator =numbers iterator();
 while(iterator hasNext())
 System.out.print(iterator nect());
 }
}
```

Which of the following statements are true?

- 1.Runtime execption will be thrown
- 2.The output is 4588null789
3. The output is 458877null789
- 4.There is a compiler error on line 1
- 5.There is a compiler error on line 7
- 6.The output is null789884577**

**444) What is the result of attempting to compile and run this program?**

```
Public class CollectionsDemo{
 Public static void main (String argv[]){
 ArrayList arrList=new ArrayList();
 ArrayList arListStr=arrList;
 ArrayList arListBuf=arrList;
 arrListStr.add(1,"SimpleString");//line6
 StringBuffer strBuff=arrListBuf.get(0);//line7
 System.out.print(strBuff toString());//line8
 }
}
```

- 1.SimpleString
- 2.Compilation fails because of an error in line6 and line8
- 3.compilation fails because of an error in line7**
- 4.null
- 5.an exception is thrown at runtime

```
Import java.util.*;

Public class SetImpl{
 Public static void main(String[] args){
 List list=new ArrayList();
 list.add("Infosys");
 list.add("Google");
 list.add("IBM");
 For(String s:list){
 System.out.println(""+s);
 }
 list.clear();
 }
}
```

}

445) What is the output?

1.It prints IBM

2.An exception occurs at runtime

3.No output

4.It prints Google

446) Which of the following statements are true if a duplicate element objT is added to a HashSet?

1.The element objT is not added and add() method returns false

2.The element objT is added successfully

3.An exception occurs during runtime

4.An exception occurs during compile time

447) Given:

Class Apple

{

A obj;

Apple(A obj){this obj=obj;}

Public A getObject(){return this obj;}

}

Class Main

{

Public static void main(String args){

//Line1

}

}

448) Which of the following code snippet can be inserted at line1 to display the output as;

76

Hello

checkbox

```
1.Apple apple=new Apple(76);
System.out.println(apple getObject());
Apple appleObj=new Apple("Hello");
System.out.println(appleObj.getObject());
2. Apple apple=new Apple(76);
System.out.println(apple getObject());
Apple appleObj=new Apple("Hello");
System.out.println(appleObj.getObject());
3. Apple apple=new Apple(76);
System.out.println(apple getObject().toString());
Apple appleObj=new Apple("Hello");
System.out.println(appleObj.toString());
4. Apple apple=new Apple(76);
System.out.println(apple getObject().toString());
Apple appleObj;
appleObj=apple;
System.out.println(appleObj.toString());
```

449) **What** will be the output of the following code;

```
Public class WrapperClassDemo {
Public static void main(string aa[])
{
Integer intwrapper=Integer.valueOf("12345");
Integer intWrapper2=Integer.valueOf("11",2);
Integer intwrwpper3=Integer.valueOf("E",16);
```

```
System.out.println(intWrapper+" "+intWrapper2+" "+intWrapper3);
```

```
}
```

```
}
```

```
1.12345 13 14
```

```
2.12345 11 14
```

```
3.12345 3 14
```

```
4.12345 3 15
```

**450) What will be the output of the following code when it is complied and executed?**

```
public class Hello{
```

```
Public static void main(String args[]){
```

```
String s="How\"are\"you?";
```

```
System.out.println(s);
```

```
}
```

```
}
```

1.The output will be

How “are” you?

2.The output will be

How\“are\”you?

3.Compilation fails

4.An exception is thrown at runtime

**451) Given :**

```
Public class TestString3{
```

```
Public static void main(String[] args){
```

```
//insert code here//Line3
System.out.println(s);
}
}
```

Which of the below code fragment when inserted independently at line 3 generate the output as 498?

- 1.String s="123456789",s=(s-"123")replace(1,3,"24")-"89";
  - 2.StringBuffer s= new StringBuffer("123456789"),s.delete(0,3),replace(1,3,"98").delete(3,8);
  - 3.StringBuffer s=new StringBuffer("123456789"),s.substring(3,6).delete(1,3).insert(1,"24")
  4. StringBuffer s=new StringBuffer("123456789"),s.substring(3,6).delete(1,2).insert(1,"24")
- StringBuilder s = new StringBuilder("123456789");  
s.delete(0,3).delete(1,3).delete(2,5).insert(1, "24");**

452) Identify which of the following class breaks its input into tokens using a whitespace pattern?

- 1.InputStreamReader
- 2.Console
- 3.Scanner**
- 4.BufferedReader
- 5.DataInputStream

453) Predict the output for the below code?

```
Public class TestDemo{
Public static void main(String[] args){
Int sum,a=10,b=0;
Try{
System.out.println(sum=a/b);
Return; //line1
```

```
}catch (ArithmeticException /Exception e) //Line 2
System.out.println(e.getMessage());
}finally{
System.out.println("In finally");
}
}
```

1.compilation fails because of the error in Line 1

2.prints:

/by zero

In finally

3.compilation fails because of the error in Line 2

4.program compiles successfully but not prints anything in console

**454)** Given:

```
Public class Excepdemo
{
Public static void main(String[] args)
{
Try
{
Method();
System.out.print("inside try");
}
Catch(RuntimeException ex)
{
System.out.print("Inside catch(RunttimeException)");
}
```

```

}

Catch(Exception ex1){

System.out.print("Inside catch(Exception)");

}

Finally

{

System.out.print("finally")

}

System.out.print("end");

}

Public static void method()

{

//Line26

}

}

```

Which code fragment can be inserted at Line26 to display the output as "Inside catch(RuntimeException)"

- 1.**throw new RuntimeException();
- 2.throw new Exception();
- 3.throws new RuntimeException();
- 4.throws new Exception();
- 5.throw new Error();

**455) What is the result of executing the following code?**

```

Package exceptions;

Public class Demo

{

Public void division(int x,int y){
```

```

Try{
 Int z=x/y;
}

Catch(exception e){
 System.out.print("Arithmetic Exception")
}

Finally{
 System.out.print("finally block")
}

}

Public static void main(String[] args)
{
 Demo demo=new Demo();
 Demo division(0,8);
}
}

```

1.Arithmetic Exception Finally block  
**2.Finally block**  
3.Arithmetic Exception  
4.An exception is thrown at runtime

**456)** Given:

```

Public class ExceptionDemo1{

Static class car implements AutoCloseable{

Public void close(){

System.out.print("Car door close")
}
}

```

```
Throw new RuntimeException(); }

Static class CarWindow implements Closeable {
 Public void close()
 {
 System.out.print("Car door close")
 Throw new RuntimeException();
 }
}
```

```
Public static void main(String[] args){
 Try{
 //line1
 }
 Catch(Exception e){
 System.out.print("catch exception");
 }
 Finally{
 System.out.print("finally");
 }
}
```

Which one of below code can be inserted at Line1 to display the output as “try block finally”

1. **Car car=new Car();**  
**CarWindow carWindow=new CarWindow();**  
**System.out.print("try block");**
2. **Car car=new Car();**  
**System.out.print("try block");**

```
3. Car car=new CarWindow();
System.out.print("try block");

4. System.out.print("try block");
```

457) Void display()

```
{
System.out.println("x= "+x+" y= "+y)
}

Public static void main(String[] args)
{
 ThisDemo thisDemo=new ThisDemo();
 thisDemo.get().display()
}
}
```

1.x=0 y=0

2.x=45 y=56

3.Compilation fails because of an error in Line 1

4.Runtime exception is thrown at line 1

458) What is the output of below code?

```
Class MyException extends Throwable{
 Public MyException(String msg){
 Super(msg);
 }
}

Public class TestDemo{
 Static void myCode() throws MyException{
 Try{
 }
```

```

Throw new MyException("Test exception")
}

Catch(Error|Exception ex){
System.out.print("Inside Error and Exception")
} }

Public static void main(String[]args)throws MyException{
Try{
myCode();
}catch(Exception ex){
System.out.print("Inside Exception")
}
}
}

1.prints "Inside Error and Exception"
2.An Exception is thrown at runtime
3.Compilation fails
4.prints "Inside Exception"

```

**459) Identify the output of the below code:**

Class ThisDemo

```

{
Int x;
Int y;
ThisDemo(){
X=45;
Y=56;
}
ThisDemo get() //Line1
{
```

Return this;

```
}

Void display()
{
 System.out.println("x=*+x+*y=*+y");

}

Public static void main(string[]args)
{
 ThisDemo thisDemo=new ThisDemo();

 thisDemo.get().display();
}

1.x=0 y=0
2.x=45 y=56
```

3.compilation fails because of an error at line 1

4.Runtime Exception is thrown at line 1

**460) Predict the output of below code**

```
Class Dog{
 Void show(){
 System.out.println("Dog");
 }
}

Class Cat{
 Void show{
 System.out.println("Cat");
 }
}
```

```

Class BullDog extends Dog{

Void show{

System.out.println("BullDog");

}

}

Public class Test{

Public static void main(String[]args){

System.out.println("Implementing type casting");

Dog d=new Dog();

BullDog bd=(BullDog)d;

Bd.show();

}

}

```

1. Displays " Implementing type casting" in console
2. Displays " Implementing type casting" and "BullDog" in console
- 3.RUNTIME ERROR: java.lang.ClassCastException**
4. Displays "BullDog" in console.

**461) What is the output of the below code?**

```

Public class Demo11{

Public static void main(String[]args){

Parent obj =new Child();

Obj.display();

}

}

Class Parent{

Public void display(int a){

System.out.println("Parent Method");

```

```
}

}

Class Child extends Parent{

Public void display()

{ System.out.println("Child Method");

}

}

1.Compilation fails // display(); not there in Parent class

2.Parent Method

3.Child method

4.An exception is thrown at runtime
```

462) Public class Demo4 extends Demo3{  
    Public Demo2 disp(){  
        //more code here  
        Return null;  
    }  
    Private String displayValue(){  
        //more code here  
    }  
}

1.Compilation of class Demo2 will fail because of an error in line 2  
2. Compilation of class Demo4 will fail because of an error in line 2  
**3. Compilation of class Demo4 will fail because of an error in line 6**  
4.Compilation will succeed for all classes and interfaces

463) Predict the output of the below code:

```
Class Employee{
```

```

//...
}

Class Manager extends Employee{
 Public void someManagerMethod(){
 //...
 }
}

Class Officer extends Employee{
 //...

Public void someMethod(Employee e){
 Manager m=(Employee)e //Line 12
 m.someManagerMethod();
}

Class Demo{
 Public static void main(String s){
 Officer obj=new officer();
 Obj.someMethod(new Officer()); //Line 19
 }
}

1.Compilation fails because of an error in Line 12 //Cannot convert Employee to Manager
2.Compilation fails because of an error in Line 19
3.Runtime exception is thrown at line 12
4. Compilation fails because of an error in Line 12 and Line 19

```

**464) Which statement is true about the classes and interfaces given below?**

```

Public interface Demo1{
 Public void display(String points);
}

```

```

Public class Demo2 implements Demo1{

Public void display(String points){};

}

Public class Demo3{

 Public Demo1 disp(){

 //more code here

 }

 Public string displayValue(){

 //more code here

 }

}

Public class Demo4 extends Demo3{

 Public Demo2 disp(){

 //more code here

 Return null;

 }

 Private String displayValue(){

 //more code here

 }

}

1.compilation of class Demo2 will fail because of an error in line2
2.compilation of class Demo4 will fail because of an error in line2
3.compilation of class Demo4 will fail because of an error in line6// cannot reduce the visibility
4.Compilation will succeed for all classes and interfaces

```

```

Public String getCustomerType()

{ return "premium";

}

```

```
1.Customer customer=new Customer(){
 Public string getCustomerType()
 {return "Premium";
 }
};
```

```
2.class C! extends Customer{
 Public string getCustomerType()
 {return "Premium";
 }
}
```

```
3.new Customer(){
 Public string getCustomerType()
 {return "Premium";
 }
}
```

**465) Predict the output of the below code:**

```
Class VarArgsDemo
{
 Static void func(int ..x)
 {
 System.out.println("Number of arguments "+x.length);
 For(int i:x)
 System.out.print(i+" ");
 System.out.println();
 }
```

```
Void func(int a) //line1
{
 System.out.println("one");
}

Public static void main(String[]args){
 New VarArgsDemo().func(150);
 Func(11,12,13,14);
 Func();
}
```

1.Number of arguments:1

150

Number of arguments:4

11 12 13 14

Number of arguments:0

2. One

Number of arguments:4

11 12 13 14

Number of arguments:0

3. Number of arguments:4

11 12 13 14

Number of arguments:0

4. Number of arguments:4

11 12 13 14

466) Given an abstract class customer below:

Public abstract class customer

{

Public abstract string getCustomertype();

}

Select a valid implementation of getCustomerType method in another class from below:

1. Abstract class C! extends Customer{

```
 Public string getCustomerType()
 {
 Return"Premium";
 }
}
```

2. Customer customer=new Customer(){

```
 Public string getCustomerType()
 {
 Return"Premium";
 }
}
```

**What will be the output for the below code**

Class Parent

```
{
 Void message()
 {
 System.out.println("Inside parent class");
 }
}
```

Class Derived extends Parent

```
{
 Void message(){
```

```
System.out.println("inside derived class");
}

Void display()
{
Message();
Super message(); //Line1
}
}

Class SuperDemo
{
 Public static void main (String args[])
{
 Derived derived=new Derived();
 Derived display(); //line2
}
}
```

1. Inside parent class

    Inside derived class

2. Inside derived class

3. Inside parent class

4. Inside derived class

    Inside parent class

**467) What will be the output of below code snippet?**

```
Package com.infy;
Class Pet{
 Public void displayName(){
 System.out.println("Inside Pet");
 }
}
```

```
Package java.pack1;
Import com.infy.pet;
Public class Dog extends pet{
 Public void displayName(){
 System.out.println("Inside Dog");
 }
}
```

```
Package java pack1;
Import com.infy.pet;
Public class Demo{
 Public static void main (String args[]){
 Pet pet=new Dog();
 Pet.displayName();
 }
}
```

- 1.inside Pet
- 2.Inside Dog
- 3.inside Pet Inside Dog

4.Compilation fails//cannot convert Dog to Pet

468) What will be the output of the following code?

```
public class test{
 Public void method()
 {
 For(int i=0;i<3;i++){
 System.out.print();
 }
 System.out.print();
 }
}
```

1. 0123
2. 012
3. Compilation fails// System.out.print should have a parameter
4. An exception is thrown at runtime

469) What are the different types of memory areas used by JVM(choose two)?

- 1.Class
- 2.Heap
- 3.Stack
- 4.Queue

470) Public class Demo{

```
Void main(){
 System.out.println("JAVA");
}
```

```
Static void main(string args){
System.out.println("spring");
}

Public static void main (String args[]){
System.out.println("hibernate");
}

Void main(Object[]args){
System.out.println("apache camel");
}
}
```

What will be the output?

1.Hibernate

2.Spring

3.JAVA

4.Apache Camel

**471) What is the output when the following snippet is compiled?**

```
Class Apple{
Int quantity;
}

Class Main{
Public static void main (String args[]){
Apple apple;
System.out.println("apple quantity");
```

```
 }
 }
```

1. 0 (Default value is printed)
2. The code will compiled successfully and prints null
3. Compilation error variable might not have been initialized
4. Apple quantity will be printed

472)

**What Is the result of the following code is compiled and executed?**

```
Class Calculator
{
Int a=123,b=200;

Public void display()
{
System.out.println("a:" +a+"b"+b+)
}
}

Class CalculatorDemo
{

Public static void main (String args[]){
 Calculator calculator1=new Calculator() //line1
 Calculator calculator2=calculator1; //line2
 calculator1.a+=1;
 calculator1.b+=1;
 System.out.println("calculator1.values");
 calculator1.display();
 System.out.println("calculator2.values");
```

```
calculator1.display();
}
}

1. calculator1.values
 a.124 b.201

calculator2.values
 a.125 b.202

2. calculator1.values
 a.124 b.201

calculator2.values
 a.123 b.200

3. calculator1.values
 a.124 b.201

calculator2.values
 a.124 b.201

4. compilation fail because of an error in line2
```

**473) JVM in java is a**

- 1.Debugger
- 2Assembler
- 3.compiler
- 4Interpreter

**474) What is the result when the following code is executed?**

```
Class Demo1{
 Public static void main (String args[]){
 Int i1=0;
```

```
Int[] j={11,111,14,19,116,215}; //line4
For (int i1:j) //line5
 System.out.print("%d",i1);
}
}
}
```

1. 11

111  
14  
19  
116  
215

2. 0

1  
2  
3  
4  
5

3. compilation fail because of an error in line5

4. Runtime exception is thrown at line 4

475) Which of the below are NOT good practices for Creating objects?

1.Lazy initialization of objects

2.Creating String literals instead of String objects

3.Creating Wrapper objects instead of primitives

4.Invoking static factory methods for immutable classes

476) Identify the issue in the below code:

```
Public class Student{
```

```
Private School school;
```

```

Private StudentDetails stuDetails;
Private Fees fees;

Public MarksHistory marksHistory(Marks marksDetails){
//computation
}
}

```

- 1.Issue: Single Responsibility principle
  - 2.Issue: Character Encoding
  - 3.Issue: Cycles between packages should be removed
- 4.Issue Lazy: Initialization**

#### 477) Given

```

Public class TestDemo{
Private static Object staticObject;
Public static Object createStaticObject(){
If(staticObject==null){
staticObject=new Object();
}
Return staticObject;
}
}

```

What changes are required in the above code for successful execution?

//As it is it gives constructor Object(int) not defined error

- 1.The method createStaticObject should be synchronized
- 2.The method createStaticObject should be private**
- 3.The staticObject reference should not be static
  - 4.The method createStaticObject should not return Object type

**478) What will happen to the following code when trying to get compiled?**

```
Class RepeatableAnnotation
{
 @SuppressWarnings("all")//line1
 @SuppressWarnings("deprecation")//line2
 Public void over()
 {
 New Date().setDate(00);
 }
}
```

1.Unreachable code error will be generated at line 2

2.Compilation will not be successful as @SuppressWarnings annotation is non-repeatable in nature

3.Warning will be issued as it is totally unnecessary to mention @SuppressWarnings("deprecation")

4.code will get compiled successfully with out any warning

**479) What will happen when the following code is subjected to compilation and execution?**

```
Interface DefaultMethodInterface1{
 Default public void defaultMethod(){
 System.out.println("DefaultMethodInterface1");
 }
}

Interface DefaultMethodInterface2{
 Default public void defaultMethod(){
 System.out.println("DefaultMethodInterface2");
 }
}

Public class TestDemo implements DefaultMethodInterface1, DefaultMethodInterface2{
```

```
Public static void main(String[] args){
 DefaultMethodInterface1 defMethIn=new TestDemo();
 defMethIn.defaultMethod();
}
}
```

- 1.An exception is thrown at runtime
- 2.Compilation fails //duplicate methods**
- 3.DefaultMethodInterface1 will get printed on the console
4. DefaultMethodInterface2 will get printed on the console

480) What is the true regarding the following code snippet?

```
Interface StaticInterface
{
 Static void staticMethod()
 {
 System.out.println("inside interface");
 }
}

Classs StaticInterfaceImpl implements staticInterface
{
 Public void staticMethod()
 {
 System.out.println("inside class");
 }
}

Public class statiDemo
{
 Public static void main(String[] args)
```

```

{
New StaticInterfaceImpl().staticMethod();
}
}

1.code will not get compiled as the static method should always be public
2.code will not get compiled as the static method is overridden in StaticInterfaceImpl
3.code will print :inside interface" on execution
4.code will print "inside class" on execution

```

**481) What happens if “default” keyword is omitted while defining a default method in interface?**

```

Interface interface1
{
Void method1()
{
System.out.println("inside default method");
}
}

a.the method cannot be overridden in the implementing classes
b.the method can be overridden in the implementing classes
c.the method cannot be given body in the interface
d.compilation error occurs

1.a and b
2.a,b and c
3.c and d// if default, then in interface , method can have body, else compilation error
4.b and c

```

**482) What will happen when the following code is compiled and executed?**

```

import java.time.LocalDate;

public class D19

```

```
{
public static void main(String[] args)
{
LocalDate date=LocalDate.of(12,11,2017);
System.out.print(date);
}
}
```

1.12 11 2017 will get printed

2.11 12 2017 will get printed

3.compilation error will be raised as the date component is not in range

**4.Execution will get raised as the date component is not in range// has to be  
LocalDate.of(year,month,dayofmonth)yyyy,mm,dd**

**483) Predict the output for the below code snippet?**

```
import java.util.Collection;
import java.util.LinkedList;

public class D18 {
 public static Collection get() {
 Collection Sorted = new LinkedList();
 Sorted.add("B");
 Sorted.add("C");
 Sorted.add("A");
 return Sorted;
 }

 public static void main(String[] args) {
 for (Object obj : get()) {
 System.out.print(obj + ",");
 }
 }
}
```

1.A,B,C

**2.B,C,A,**

3.compilation fails

4.The code runs with n output

**484) Which of the following statements are true if a duplicate element objT is added to a HashSet?**

**1.The element objT is not added and add() method returns false**

- 2.The element objT is added successfully
- 3.An exception occurs during runtime
- 4.An exception occurs during compile time

**485) Which of these statements compile?(choose at least one)**

checkbox

- 1.HashSet hs=new HashSet();
- 2. HashSet set=new HashSet();
- 3.List list=new Vector();

List values=new HashSet();// Cannot convert hashset to List

List objects=new ArrayList();

Map hm=new HashMap();

**486) What will happen when the following code is executed?**

```
import java.util.*;

public class D16 {
 public static void main(String[] args) {
 List List1 = new ArrayList();
 List1.add("1");
 System.out.println("");
 List1.add("2");
 List1.add("3");
 List list2 = new LinkedList(List1);
 List1.add(list2);
 list2 = List1.subList(1, 2);
 list2.clear();
 System.out.print(List1 + "");
 }
}
```

[1, 3, [1, 2, 3]]

- 1.the program compiles successfully and throws exception during runtime
- 2.the program has compilation error
- 3.the program compiles successfully and executes without displaying anything
- 4.the program compiles successfully and displays 1 2 3 as output

**487) What is the result of attempting to compile and run this program?**

```
import java.util.ArrayList;

public class D15 {
 public static void main(String argv[]){
 ArrayList arrList=new ArrayList();
 ArrayList arrListStr=arrList;
 ArrayList arrListBuf=arrList;
 arrListStr.add(1, "SimpleString"); //line6
 String strBuff=arrListBuf.get(0); //line7
 System.out.println(strBuff.toString()); //line8
 }
}
```

1.simpleString

2.compilation fails because of an error in line6 and line8

**3.compilation fails because of an error in line 7 //cannot convert object to String**

4.null

**488) What will be the output of the following code?**

```
public class D14 {
 public static void main(String args[]) {
 Integer intWrapper = Integer.valueOf("12345");
 Integer intWrapper2 = Integer.valueOf("11", 2);
 Integer intWrapper3 = Integer.valueOf("E", 16);
 System.out.println(intWrapper + " " + intWrapper2 + " " + intWrapper3);
 }
}
```

1.12345 13 14

2.12345 11 14

**3.12345 3 14**

4.12345 3 15

**489) What is the result if we compile and execute the below code?**

```
public class D13 {
 public static void main(String[] args) {
 String joinString = String.join(".", "java", "programming", "course");
 String s1 = "JAVA", s2 = "java", s3 = "Java";
 s1.toLowerCase();
 s3 = s3.replace("J", "j");
 System.out.println(joinString);
```

```
 System.out.println(s1.equals(s2) + "," + (s2 == s3));
 }
}
Java.programming.course
False,false
```

1.java:programming:course

False,false

2. java:programming:course

False,true

3. java:programming:course

True,true

4. java:programming:course

False,false

**490) Predict the output of the below code;**

```
public class D12 {
 public static void main(String[] args) {
 String value1 = "Hello";
 String value2 = new String("Hello");
 System.out.println(value1.equals(value2) + "," + (value1 == value2));
 String value3 = value2.intern();
 System.out.println((value1 == value3) + "," + value1.equals(value3));
 }
}
```

1.false,true

True,true

2. true,false

true,false

3. true,false

True,true

4. false,true

false,true

**491) What is the output when the below code is compiled and executed?**

```
Package exceptions;
Public class Demo
{
 Public static void main(String[] args)
 {
 Try
 {
 Return;
 }
 Finally
 {
 System.out.println("finally");
 }
 }
}
```

**1.Finally**

- 2.compilation fails
- 3.the code runs with no output
- 4.an exception is thrown at runtime

**492) Given:**

```
Public class Teststring3{
 Public static void main(String[] args){
 //insert code here//libne3
 System.out.println(s);
 }
}
```

Which of the below code fragment when inserted independently at line3,generate the output as498?

- 1.String s="123456789",s=(s-"123") replace(1,3,"24")-"89";
- 2.StringBuffer s=new StringBuffer("123456789"),s.delete(0,3).replace(1,3,"98").delete(3,8);
- 3.StringBuffer s=new StringBuffer("123456789"),s.substring(3,6).delete(1,3).insert(1,"24");
- 4.StringBuffer s=new StringBuffer("123456789"),s.substring(3,6).delete(1,2).insert(1,"24");

493) Public class Fork{

```
 Public static void main(string[]args){
 If(args.length==1| args[1].equals("test")){
 System.out.println("test case");
 }
 Else{
 System.out.println("production"+args[0]);
 }
 }
}
```

What is the result when we execute the command-line invocation as java?

- 1.test case
- 2.production java
- 3.test case live2
- 4.compilation fails
- 5.an exception is thrown at runtime**

494) Given that MyNewTest.java is a test case with one test method

```
//Assume all the required imports are added
Public class MyNewTest {
```

```
@Test
Public void test1(){
//
}
}
```

Junit Testsuite.java is a test suite with two test classes FirstTestClass and SecondClass with one test method

```
@RunWith(Suite.class)
@Ignore
@Suite.SuiteClass({
 FirstTestClass.class,SecondTestClass.class,})
```

```
Public class JunitTestSuite{
}
```

JunitTestSuite1.java is a Test Suite defined as follows;

```
@RunWith(Suite.class)
@Suite.SuiteClass({
 JunitTestSuite.class,
 MyNewTest.class})
```

```
Public class JunitTestSuite1{
}
```

How many tests will run when JunitTestSuite1 is run?

- 1) 1
- 2) 2
- 3) 3
- 4) Compilation fails

495) Predict the output for the below code?

```
//Assume all the required imports are added
```

---

### Loopdemo.java

---

```
Public class display(int marks){
 If (marks>=85&&marks<99){
 Return 'A'; }
 Else if (marks>65&& marks<=84){
 Return 'B'; }
 Else{
 Return 'F';} } }
```

---

### TestLoop.java

---

```
public class Testloop{
 Loopdemo tc=new Loopdemo();
 Char value;

 @Test

 Public void testdisplay(){
 Value=tc.display(80);
 Assert.assertSame(Value,'B');
 }
}
```

- 1.Out of 8 branches branch coverage of the above will be 3
- 2.Out of 7 branches branch coverage of the above will be 2
- 3.Out of 5 branches branch coverage of the above will be 1
- 4.Out of 5 branches branch coverage of the above will be 2

#### **496) What is the result when we execute the command line invocation as java ?**

```
Public class Forks{
 Public static void main (String args[])
 If(args.length==1|args[1].equals("test")){
 System.out.println("test case");
```

```
 }
}

Else{
 System.out.println("production"+args[0]);
}
}
```

- 1. test case
- 2.production java
- 3.test case live2
- 4.compilation fails

**5.An exception is thrown at runtime**

**497) What is the output of the below code?**

```
//Assume all the required import statements are added
public class TestClass
{
 @Test
 public void test(){
 String a="";
 assertEquals("a",a);
 }
}
```

- 1.Test Passes
- 2.Test fails**
- 3. An exception thrown at runtime
- 4.Compilation fails

**498) Consider the below code snippet:**

```
Locale locale=new Locale("da","DK");
```

```
NumberFormat numberFormat=NumberFormat.getInstance(locale);
String number=numberFormat.format(100,99);
System.out.println(number);
```

Here NumberFormat.getInstance() follows which design pattern?

**1.Factory method pattern**

- 2.singleton pattern
- 3.Abstract factory pattern
- 4.Builder pattern

499) Which of the below statements are true about design patterns?

- 1.There are three design patterns defined for java
- 2.We can use each design pattern only once per application
- 3.Design patterns are conceptual reusable solution

1.Statements 1,2,3

2.statements 3,2

3.only statement 2

4.only statement 3

500) What change need to be made in the following code to make the singleton pattern correct?(choose that all apply)

```
Public class Employee{
 Public static Employee C;
 Private Employee() {}
 Public static Employee get Employee()
 {
```

```
If(employeeInstance==null){
 employeeInstance=new Employee();
}

Return employeeInstance;
}

}

1.None the singleton pattern is properly implemented
2.Rename employee to instance
3. Rename getEmployee() to getInstance()
4.change the access modifier of employeeInstance from public to private
5.mark employee final
```

\*\*\*\*\*Set 1\*\*\*\*\*

501) Qn 1 :

```
enum Employee{
 private TOP,
 public MEDIUM,
 protected BOTTOM;
}
```

ANSWER -->

Compilation Error due to private TOP

Compilation Error due to public MEDIUM

Compilation Error due to protected BOTTOM

=====

502) Qn 2 : (#Rep-2)

```
Class Customer{
 public void display(){}
}

Class Employee{
 public void display(){}
}

Class Demo{
 public void main(String[] args){
 Customer customer = new pare
 Employee employee = (Employee) customer;//Line1
 employee.display();//Line2
 }
}
```

```
}
```

ANSWER --> Compilation Fails because of an error in Line1

---

503) Qn 4 :(#Rep)

pattern matching for the following

false

Simple

demo

for

regular

expressions

using

pattern

matching

Simple demo for regular expressions using pattern matching

```
public class RegExDemo1{
 public static final String string1 = "Simple demo for "+"regular expressions "+"using pattern matching.";
 public static void main(String[] args){
 //Line1
 //Line2
 }
}
```

ANSWER --> 1

```
System.out.println(string1.matches("\\r.*"));
String[] splitString = (string1.split("\\s+"));
for(String string:splitString){
 System.out.println(string);
}
System.out.println(string1.replace("\\s+","\t"));
```

=====

504) Qn 5 :

which will compil successfully?

```
int number = 1_234; //CORRECT
double d1 = 1_234_.0;
double d2 = 1_234._0;
double d3 = 1_234.0_;
long num = 1_000_00; //CORRECT
```

=====

505) Qn 6 :

```
public class Main{
public static void main(String args[]){
int twoD[][] = new int[4][];
int twoD[0] = new int[1];
int twoD[1] = new int[2];
int twoD[2] = new int[3];
```

```
int twoD[3] = new int[4];
for(int i = 0;i<4;i++){
 for(int j = 0;j<i+1;j++){
 twoD[i][j]=i+j;
 }
}

for(int i = 0;i<4;i++){
 for(int j = 0;j<i+1;j++){
 twoD[i][j];
 }
 System.out.println();
}
```

ANSWER --> Compilation Fails

---

506) Qn 7 :(#Rep)

```
public class Test{
 public static void main(String[] args){
 int[][]x;
 x=new int[3][4];
 for(int i =0;i<3;i+=2){
 for(int j = 0; j<4; j++){
 x[i][j]=i+j;
 }
 System.out.println(x[i][j]+" ");
 }
 }
}
```

```
 }
 }
 }
```

ANSWER --> 0 1 2 3 2 3 4 5

---

=====

507) Qn 8 : (#Rep)

```
enum Fruits{
```

```
APPLE,
```

```
MANGO,
```

```
STRAWBERRY,
```

```
LICHI;
```

```
double calculate(double a, double b){
switch(this){
case APPLE:
return a+b;
case MANGO:
return a -b;
case STRAWBERRY:
return a*b;
case LICHI:
return a/b;
default:
throw new AssertionError("Unknown Input. "+this);
}
}
```

```
}

public static void main(String[] args){
//Line3(To print 188.22)
}

}
```

ANSWER --> c) double res=Fruits.MANGO.calculate(298, 109.78);//Third One

```
 System.out.println(res);
```

```
=====
```

508) Qn 9 : (#Rep - 2)

```
class Demo2{

public static void main(String args[]){
int[] x = {111,112,113,114,115,116,117,118,119,110};
//Line1(to print all the x values)
System.out.println("Count is"+i);
}
}
}
```

ANSWER --> for(int i=0;i<=x.length;i++){//it prints the count

```
 for(int i:x) // prints all the values
```

```
=====
```

509) Qn 10:(#Rep)

```
import java.util.regex.Pattern;
```

```
public class RegexDemo2{
 private static final String String1 = ":";
 private static final String String2 = "one:two:three:four:five";

 public static void main(String[] args){
 Pattern pattern = Pattern.compile(String1);//Line1
 String[] strArr = pattern.split(String2);//Line2
 for(String str:strArr){
 System.out.println(str);
 }
 }
}
```

ANSWER -->

one  
two  
three  
four  
five

\*\*\*\*\*Set  
2\*\*\*\*\*

510) Qn 01:

```
class Demo1{
 int a = 11;
 void display1(){
 Demo2 demo2 = new Demo2();
 demo2.display2();
 getValues();
 }
}
```

```
class Demo2{
 int b = 32;
 void display2(){
 System.out.print(a+" ");//Line1
 }
}
void getValues(){
 System.out.println(b); //Line 2
}
int b = 78;
}

class Demo11{
 public static void main(String[] args){
 Demo1 demo1 = new Demo1();
 demo1.display1(); //Line 3
 }
}
```

ANS: 11 78

---

511) Qn 03:

```
public class Demo1{
 public static void main(String... args){
 Pattern pattern = Pattern.compile("x*y");
 Matcher match = pattern.matcher("y");
 boolean boolean1 = match.matches();
 }
}
```

```
System.out.println(boolean1);
}
}
```

ANS: True

---

512) Qn 06: (#Rep)

```
enum customer{
 private CUSTID;
 public CUSTNAME;
 protected ADDRESS;
}
```

ANS: Compilation fails

---

513) Qn 07:(#Rep)

```
public class OperatorDemo{
 public static void main(String[] args){
 int i1 = 15;
 String b1 = (i1>30)? "Red":(i1>20)? "Green":(i1>10)? "Blue": "Violet";
 System.out.println(b1);
 }
}
```

Ans: Blue

=====

514) Qn 09:(#Rep)

```
public class Test{
 public void method(){
 for(int i = 0;i<3;i++){
 System.out.print(i);
 }
 }

 public static void main(String args[]){
 method();
 }
}
```

Ans: Compilation Fails

=====

515) Qn 03:

Enums can be defined inside..... CHECK BOXES....

- a) An interface
- b) A class
- c) A static context
- d) A method

ANSWER --> a,b,c

=====

516) Qn 06:

```
public class Test{
 public void method(){
 for(int i = 0;i<3;i++){
 System.out.println(i);
 }
 System.out.println(i);
 }
}
```

Answer --> Compilation fails

=====

517) Qn 09:

```
public class Fork{
 public static void main(String args[]){
 if(args.length == 1 | args[1].equals("test")){
 System.out.println("test case");
 }
 else{
 System.out.println("production"+args[0]);
 }
 }
}
```

what is the result when we execute the command line as : java Fork live2

- a) Test CAsE
- b) Production java
- c) test case live2
- d) Compilation fails
- e) An exception thrown at run time

ANSWER --> An exception thrown at run time

=====

\*\*\*\*\*Set  
1\*\*\*\*\*

518) Qn 11:(Rep)

```
public abstract class customer{
 public abstract String getCustomerType();
}
valid implementation of abs method in another class
```

ANSWERS -->

```
a) abstract class Demo extends Customer{
 public String getCustomerType(){
 return "Premium";
 }
}
```

```
c) class Demo extends Customer{
 public String getCustomerType(){
 return "Premium";

 }
}
```

=====

519) Qn 12:

```
interface Fruits{
 public void printPrice();
}

1.public class Apple{
2.public static void main(String[] args){
3.Fruits fruits = new Fruits(){
4.public void printPrice(){
5.System.out.println("150");
6.}
7.}
8.fruits.printPrice();
9/}
10.}
```

ANSWER --> 150

=====

520) Qn 13:

```
abstract class A{
 public A(){
 System.out.println("First");
 }
 abstract void method();
}
```

```
class B extends A{
 public B(){
 System.out.println("Second");
 }
```

```
@Override
void method(){
 System.out.println("inside abstract methos");
}
}

public class AbstractClassDemo{
 public static void main(String[] args){
 //Line23
 }
}
```

To get output >>

```
First
Second
Inside abstract method
```

ANSWER -->

c) A a = new B();

a.method();

d) B b = new B();

b.method();

=====

521) Qn 14 :

```
public class Developer{
}

public class Employee{
 public String empName;
}

public class Tester extends Employee{
 public Developer developer;
}

public class Testing extends Tester
}
```

which are true>>

Testing is

1. has a empName
2. testing has a Developer//correct
3. is a Developer
4. Testing is a Employee//correct

5. tester is a testing

6. Employee has a Developer

=====

522) Qn 15 :(# Rep)

```
class Mammal{
 String name = "furry";
 String makeNoise(){
 return "Generic Noise";
 }
}

class Zebra extends Mammal{
 String name = "Stripes";
 String makeNoise(){
 return "Bray";
 }
}

public class Demo11{
 public static void main(String[] args){
 new Demo11().go();
 }
 void go(){
 Mammal m = new Zebra();
 System.out.println(m.name+" "+m.makeNoise());
 }
}
```

ANSWER --> furryBray

=====

523) Qn 16:

```
class Customer{
 int customerId = 11201;

 Customer(){
 customerId = 11240;
 }
}

class Main(
 public static void main(String[] args){
 Customer customer = new Customer();
 System.out.println(customer.customerId);
 }
}
```

ANSWER --> 11240

=====

Qn 17:(#Rep)

```
class ThisDemo{
 int x;
 int y;

 This Demo(){
 x=45;
 y=56;
 }
}
```

```
ThisDemo get()//Line1
{
 return this;
}

void display(){
 System.out.println("x="+x+"y="+y);
}

public static void main (String[] args){
 ThisDemo thisDemo = new ThisDemo();
 thisDemo.get().display();
}
}
```

ANSWER --> x=45y=56

=====

524) Qn 18:(#Rep)

```
class Apple{
 private Apple()//Line
 {
 System.out.println("Apple Constructor");
 }

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

public class Main{
 public static void main(String args[]){
}
```

```
Apple apple = new Apple();//Line2
apple.display();
}
}
```

ANSWER --> Unresolved Compilation problem. the constructor Apple() is not visible

---

525) Qn 19:

```
class Trainer{
 public void display(String name){
 System.out.println("Am a trainer..");
 print(name);
 }
 public void print(String name){
 System.out.println("I train"+name+".");
 }
}

public class Trainee extends Trainer{
 String myname;

 public Trainee(String myname){
 super();
 this.myname = myname;
 }
 public void display(String name){
 super.display(name);
 System.out.println("Am a trainee...");
 }
}
```

```
print("Java");

}

public void print(String name){

super.print(name);

System.out.println("I want to learn:"+name);

}

public static void main(String args[]){

Trainer trainee = new Trainee("XYZ");

trainee.display("Java");

}

}
```

ANSWER --> Am a trainer..

I trainJava.

I want to learn:Java

Am a trainee...

I trainJava.

I want to learn:Java

=====

526) Qn 20:

```
interface Fruits{

String fname = "Pomegranate";

String getName(String name);

}

public class Apple implements Fruits{

public String getName(String name){

System.out.println("Inside Apple class");

}
```

```
return "Fruit from interface is:"+fname+"and fruit from class is:"+name+".";
}

public static void main(String args[]){
 Apple apple = new Apple();
 System.out.println(apple.getName("Guava")+" ");
 apple.getName("Guava");
}
}
```

aNSWER --> Inside Apple class

Fruit from interface is:Pomegranateand fruit from class is:Guava.

Inside Apple class

```
*****Set
2*****
```

527) Qn 01:(Rep - 2)

Employee.java

```
package com.infy;
```

```
public class Employee{
 static final int empid = 1101;
}
```

SuperDemo.java

```
import com.infy.Employee;
```

```
class Unit extends Employee{
 int empid = 1102;
 void display(){
 //Line 7
 }
}
```

```
}

}

class SuperDemo{
 public static void main(String[] args)
 {
 Unit unit = new Unit();
 unit.display();
 }
}
```

which will fail @ Line 7?(Checkboxes)

- a) System.out.println("Maximum speed: " +super.empId);
- b) System.out.println("Maximum speed: " +new Employee().empId);
- c) Employee emp1 = new Employee();  
System.out.println("Maximum speed: " +new Unit().empId);
- d) System.out.println("Maximum speed: " +Employee.empId);

Ans: only c (Prints 1102) ,all other options fail

=====

528) Qn 03:(#Rep)

```
class Apple{
 private int quantity = 40;
 private int getQuantity(){
 return quantity;
 }
```

```
public void setQuantity(int quantity){
 this.quantity = quantity;
}

void display(){
 System.out.println("Inside Apple");
 getQuantity();
}
}

public class AppleDemo{
 public static void main(String[] args){
 Apple apple = new Apple();
 apple.display();
 }
}
```

ANS: Inside Apple

---

529) Qn 04:(#Rep)

```
public class Demo11{
 public static void main(String[] args){
 Parent obj = new Child();
 obj.display();
 }
}

class Parent{
 public void display(int a){
 System.out.println("Parent Method");
 }
}
```

```
}

class Child extends Parent{

public void display(){

System.out.println("Child method");

}

}
```

ANS: Compilation Fails

=====

530) Qn 05:

```
public class Game{

public static void main(String[] args){

displayRegistration("Hockey");//Line 1

displayRegistration("Kho-Kho",132,102,36);//Line 2

}

public static void displayRegistration(String gameName, int... id){

System.out.println("Registartion for"+gameName+":");

for(int i = 0;i<id.length;i++){

System.out.println(id[i]+ " ");

}

}

}
```

Ans: Registration for Hockey:

Registration for Kho-Kho:

132 102 36

=====

531) Qn 06:

```
class Parent{
}

class Child extends Parent{
}

final class GrandChild extends Child{
}
```

which is false?

- a) represents multi-level inheritance with two levels
- b) GrandChild can access the protected and public members of Parent & Child classes
- c) Instance of Parent class can accept the reference of Child class but not the reference of GrandChild class
- d) The GrandChild class can override the methods of Parent and child classes

ANSWER --> c

=====

532) Qn 08:

```
class VarArgsDemo{
}

static void func(int ...x){
}

System.out.println("Number of arguments:"+x.length);

for(int i:x)
System.out.print(i+" ");

System.out.println();
```

```
}

void func(int a)//Line 1

{
System.out.println("one");

}

public static void main(String args[]){
new VarArgsDemo().func(150);
func(11,12,13,14);
func();
}
}
```

ANS: one

Number of arguments:4

11 12 13 14

Number of arguments:0

---

533) Qn 09:

```
class Employee{

void disp(char c){

System.out.print("Employee name starts with:"+c+":");

System.out.print("His experience is : 11 years. ");

}

}

class Main extends Employee{

void disp(char c){

super.disp(c);

}}
```

```

System.out.println("Another employee name also starts with :" + c + ":");

new Employee().disp('D');

disp(7);

}

String disp(int c){

System.out.println("His experience is :" + c + "years.");

return "Bye";

}

}

public class Demo11{

public static void main(String[] args){

Employee emp = new Main();

emp.disp('S');

}

}

```

ANS: Employee name starts with:S.His experience is : 11 years. Another employee name also starts with :S.

Employee name starts with:D.His experience is : 11 years. His experience is :7years.

---

534) Qn 10:(#Rep)

```

public class BasePlant{

private String name;

public BasePlant(String name){

this.name = name;

}

public String getName(){


```

```
return name;
}
}

public class Tree extends BasePlant{
 public void growFruit(){}
 public void dropLeaves(){}
}
```

which is true?(Check boxes)

- a) will be compiled successfully
- b) will be compiled if "public Tree(){super("Plant");}" is added to Tree class
- c) will be compiled if "public BasePlant(){Tree();}" is added to BasePlant class
- d) will be compiled if "public BasePlant(){ this("Plant");}" to BasePlant class

ANSWER - b,d

\*\*\*\*\*Set  
3\*\*\*\*\*

535) Qn 02:

```
class Student{
 int regNo;
 String name;
 float fee;
 Student(int regNo,String name,float fee){
 regNo = regNo;
 name = name;
 fee = fee;
 }
```

```
void display(){
System.out.println(regNo+" "+name+" "+fee);}
}

public class Main{
public static void main(String args[]){
Student student1 = new Student(1101,"Jack", 6500f);
Student student2 = new Student(1102,"John", 7000f);
student1.display();
student2.display();
}
}
```

ANSWER -->

0 null 0.0

0 null 0.0

---

536) Qn 04:(#Rep)

```
abstract class Customer
{
public int custId;
Customer()
{
custId = 23456;
}
abstract public void setId();
abstract final public void getId(); // comp error
}
```

```
class IDemo extends Customer // comp error
{
 public void setId(int custId)
 {
 this.custId = custId;
 }

 final public void getId() //comp error
 {
 System.out.println("CustomerId: "+custId);
 }

 public static void main(String args[])
 {
 IDemo demo = new IDemo();
 demo.setId(11012);
 demo.getId();
 }
}
```

- a) Compilation fails @ Line 09
- b) Compilation fails @ Line 11
- c) Runtime Exception @ Line 17
- d) CustomerId:11012
- e) Compilation fails @ Line 17

ANSWER --> a,b,e

=====

```

*****Set
1*****

537) Qn 21:(#Rep-2)

public class ExcepDemo{

public static void main(String args[]){

try{
method();
System.out.println("Inside try");
}

catch(RuntimeException ex){
System.out.println("Inside catch(RuntimeException)");
}

catch(Exception ex1){
System.out.println("Inside catch(Exception)");
}

finally{
System.out.println("finally");
}

System.out.println("end");
}

public static void method(String args[]){
//Line 26(which throw statement to display the output "Inside catch(Runtime Exception) finally
end"??????)
}

}

```

ANSWER --> throw new RuntimeException();

538) Qn 22:

```
public class TestDemo{
 public static void main(String args[]){
 int sum,a=10,b=10;
 try{
 System.out.println(sum = a/b);
 return;//Line 1
 }catch(ArithmeticException | Exception e){//Line 2
 System.out.println(e.getMessage());
 }
 finally{
 System.out.println("finally");
 }
 }
}
```

ANSWER --> Compilation fails due to the error in line 2

539) Qn 23:

```
public class A{
 public int display(String str, Integer... data) throws ArrayIndexOutOfBoundsException{
 String signature = "(String, Integer[])";
 System.out.println(str + " " + signature);
 }
}
```

```
return 1;
}
}

public class B extends A{
 public int display(String str, Integer... data) throws Exception{
 String signature = "(String.Integer[])";
 System.out.println("Overridden:"+str+" "+signature);
 return 0;
 }
}

public static void main(String[] args){
 B sb = new B();
 try{
 sb.display("hello",3);
 }catch(Exception e){
 }
}
}
```

aNSWER --> Compilation fails

```
=====
=====
```

540) Qn 24:(#Rep)

```
class MyException extends Throwable{
 public MyExceptions(String msg){
 super(msg);
 }
}
```

```
public class TestDemo{
 static void myCode() throws MyException{
 try{
 throw new MyException("Test exception");
 }catch(Error|Exception ex){
 System.out.println("inside Error and Exception");
 }
 }

 public static void main(String[] args) throws MyException{
 try{
 myCode();
 }
 catch(Exception ex){
 System.out.println("Inside Exception");
 }
 }
}
```

aNSWER --> Exception @ runtime

=====

541) Qn 25:

```
package exceptions;

public class Demo{
 public void division(int x, int y){
 try{
 int z = x/y;
```

```
}catch(Exception e){
System.out.println("arithmetic exception");
}finally{
System.out.println("Finally block");
}
}

public static void main(String[] args){
Demo demo = new Demo();
demo.division(0,8);
}
}
```

ANSWER --> Finally block

```
*****Set
2*****
```

542) Qn 01:

```
public class ExceptionInClass{
int data = 10;
void calculate() throws Exception{
try{
data++;
try{
data++;
//Line12
}
catch(Exception ex){
data++;
}
}
```

```
void display(){
 System.out.println(data);
}

public static void main(String[] args) throws Exception{
 ExceptionInClass exceptionInClass = new ExceptionInClass();
 exceptionInClass.calculate();
 exceptionInClass.display();
}
}
```

which code has to be inserted in Line 12 to display output as 15

a) try{  
 data++;  
 throw new Exception();  
}  
catch(Exception ex){  
 data++;  
 throw new Exception();  
}

b)try{  
 data++;  
 throw new Exception();  
}  
catch(Exception ex){  
}

```
c) try{
 data++;
 throw new RunTimeException();
}
catch(Exception ex){
 data++;
 throw new RunTimeException();
}
```

```
d) try{
 throw new Exception();
}
catch(Exception ex){
 data--;
 throw new Exception();
}
```

ANSWER --> a

=====

543) Qn 02:

```
public class TestDemo{
 public static void main(String[] args){
 int x = 20;
 int y = 2;
 try{
```

```
for(int z = 4;z>=0;z--){
 int ans = x/z;
 System.out.println(ans+" ");
}
}catch(Exception e1){
 System.out.println("E1");
}
}catch(ArithmeticException e2){
 System.out.println("E2");
}
}
```

ANSWER --> Compilation fails

---

544) Qn 03:(#Rep)

```
class ExDemo{
 public static void main(String[] args){
 try{
 throw 110;
 }
 catch(int ex){
 System.out.println("Caught Exception"+ex);
 }
 }
}
```

ANSWER --> Compilation fails

```
=====
=====
```

545) Qn 04:

```
public class TestDemo{
 public static void main(String args[]){
 for(int a=0;a<10;++a){
 try{
 if(a%3 == 0)
 throw new Exception("Except1");
 try{
 if(a%3 == 1)
 throw new Exception("Except2");
 System.out.println(a);
 }catch(Exception inside){
 a*=2;
 }finally{
 ++a;
 }
 }catch(Exception outside){
 a+=3;
 }finally{
 ++a;
 }
 }
 }
}
```

ANSWER --> 5

8

---

```
*****Set
1*****
```

546) Qn 26:(#Rep-2)

```
public class WrapperClassDemo{
 public static void main(String ass[]){
 int x=90;
 Integer i1 = new Integer(x);
 int y = 90;
 Integer i2 = new Integer(y);
 System.out.println(i1.compareTo(i2)+" "+Integer.compare(i2,i1)+" "+i1.equals(i2)+" "+(i1==i2));
 }
}
```

which method compares the given values and return an int, which tells whether the given vals are equal / greater/ lesser?

ANSWER --> compareTo(), compare()

---

---

547) Qn 27:

```
public class TestDemo{
 public static void main(String args[]){
 Integer n1 = new Integer(100);
 Integer n2 = new Integer(100);
```

```
Integer n3 = 127;
Integer n4 = 127;
Integer n5 = 128;
Integer n6 = 128;
int n7 = 129;
int n8 = 129;

System.out.println(n1==n2);
System.out.println(n3==n4);
System.out.println(n5==n6);
System.out.println(n7==n8);
}
}
```

ANSWER -->

false  
false  
false  
true

=====

548) Qn 28:(#Rep)

```
public class TestDemo{

public static void main(String args[]){

String value1 = "Hello";

String value2 = new String("Hello");

System.out.println(value1.equals(value2)+","+(value1==value2));

String value3 = value2.intern();
```

```
System.out.println(value1==value3)+"."+value1.equals(value3));
}
}
```

ANSWER -->

true,false

true,true

```
=====
```

549) Qn 29:

```
public class StringTest{
 public static void main(String args[]){
 String joinString = String.join(".", "java", "programming", "course");
 String s1 = "JAVA", s2 = "java", s3 = "Java";
 s1.toLowerCase();
 s3 = s3.replace('J', 'j');
 System.out.println(joinString);
 System.out.println((s1.equals(s2))+"."+(s2==s3));
 }
}
```

ANSWER -->

java.programming.course

false,false

```
*****Set
2*****
```

550) Qn 01:

```
public class CalendarClass{
 public static void main(String args[]){
 Calendar calendar = Calendar.getInstance();
 //Line1
 System.out.println("Maximum number of weeks in month:"+maximum);
 //Line 2
 System.out.println("MAximum number of days in year:"+maximum);
 }
}
```

which code to be inserted to get output weeks- 5 and days - 366

- a)int maximum = calendar.getMaximum(Calendar.WEEK\_OF\_MONTH);  
maximum = calendar.getMaximum(Calendar.DAY\_OF\_YEAR);
- b)int maximum = calendar.getActualMaximum(Calendar.WEEK\_OF\_MONTH);  
maximum = calendar.getLeastMaximum(Calendar.DAY\_OF\_YEAR);
- c)int maximum = calendar.getLeastMaximum(Calendar.WEEK\_OF\_MONTH);  
maximum = calendar.getLeastMaximum(Calendar.DAY\_OF\_YEAR);
- d)int maximum = calendar.getActualMaximum(Calendar.WEEK\_OF\_MONTH);  
maximum = calendar.getMaximum(Calendar.DAY\_OF\_YEAR);

ANSWER --> d

551) Qn 02:(#Rep)

```
public class Hello{
 public static void main(String args[]){
 String s = "How\"are\"you?";
 System.out.println(s);
 }
}
```

ANSWER --> How"are"you?

Qn 03:(#Rep)

Which class breaks its input into tokens using a whitespace pattern?

- a) `InputStreamReader`
  - b) `Console`
  - c) `Scanner`
  - d) `BufferedReader`
  - e) `DataInputStream`

ANSWER --> Scanner

552) Qn 04:(#Rep)

```
public class WrapperClassDemo{
 public static void main(String aa[]){
 Integer intWrapper = Integer.valueOf("12345");
 Integer intWrapper2 = Integer.valueOf("11",2);
 Integer intWrapper3 = Integer.valueOf("E",16);
 System.out.println(intWrapper+ " "+intWrapper2+" "+intWrapper3);
 }
}
```

ANSWER --> 12345 3 14

```
*****Set
3*****
```

ALL REPEATED

```
*****Set
4*****
```

ALL REPEATED

```
*****Set
1*****
```

553) Qn 30:

```
public class Group extends TreeSet{
 public static void main(String args[]){
 Group g = new Group();
 g.add(new Person("Hans"));
 g.add(new Person("Jane"));
 g.add(new Person("Hans"));
 System.out.println("Total"+g.size());
 }
}
```

```
public boolean add(Object o){
 System.out.println("adding:"+o);
 return super.add(o);
}
}

class Person{
 private final String name;
 public Person(String name){
 this.name=name;
 }
}
```

ANSWER --> Adding Hans

An exception occur @ runtime

---

554) Qn 31:

which will be compiled?

```
HashSet hs = new HashSet();
HashSet set= new HashSet();
List list = new Vector();
List values = new HashSet(); ---> WRONG
List objects = new ArrayList();
Map hm = new HashMap();
```

---

555) Qn 32:(#Rep)

```
import java.util.*;

public class Demo13 implements Comparable, Comparator{

private int number;

private String data;

Demo13(int i, String str){

this.number = i;

this.data = str;

}

public String toString(){

return ""+number;

}

public int compareTo(Demo13 demo){

return data.compareTo(demo.data);

}

public int compare(Demo13 demo1, Demo13 demo2){

return demo1.number - demo2.number;

}

public static void main(String[] args){

Demo13 demo1 = new Demo13(88,"a");

Demo13 demo2 = new Demo13(55,"b");

//Line4

}

}

To produce output [89,45][45,89]
```

ANSWER --> Compilation fails

=====

556) Qn 33 :(#Rep)

```
public class TestDemo{
 public static void main(String[] args){
 TreeSet tset = new TreeSet();
 tset.add(new Item());
 TreeSet b = tset;
 }
}
```

aNSWER --> Compilation fails

---

557) Qn 34:

```
public class TestDemo{
 public static void main(String[] args){
 ArrayList strings = new ArrayList();
 strings.add("aAaA");
 strings.add("AaA");
 strings.add("aAa");
 strings.add("AAaa");
 Collections.sort(strings);
 for(String string:strings){
 System.out.println(string);
 }
 }
}
```

ANSWER --> Compilation fails

\*\*\*\*\*Set 2\*\*\*\*\*

558) Qn 02:

```
import java.util.HashSet;

public class TestDemo{
 public static void main(String... sss){
 HashSet myMap = new HashSet();
 String s1 = new String("das");
 String s2 = new String("das");
 NameBean s3 = new NameBean("abcdef");
 NameBean s4 = new NameBean("abcdef");
 myMap.add(s1);
 myMap.add(s2);
 myMap.add(s3);
 myMap.add(s4);
 System.out.println(myMap);
 }
}

class NameBean{
 private String str;
 NameBean(String str){
 this.str = str;
 }
 public String toString(){
 return str;
 }
}
```

ANSWER -> [das, abcdef, abcdef]

=====

559) Qn 03:

```
public class Demo{
 public static void main(String args[]){
 List arrList = new ArrayList();
 arrList.add("First");
 arrList.add("Second");
 arrList.add(23);
 for(String str:arrList)
 System.out.println(str);
 }
}
```

ANSWER --> c and d which are as follows

```
List arrList = new ArrayList();
arrList.add("First");
arrList.add("Second");
arrList.add("23");
for(String str:arrList)
 System.out.print(str);
=====
```

560) Qn 04:(#Rep)

```
public class TestDemo{
 public static void main(String args[]){
 Set set = new TreeSet();
 set.add(1);//Line 1
 set.add(2.7);//Line 2
```

```
set.add(2);//Line 3

for(Object element:set){
 System.out.println(element);
}

}
```

ANSWER --> An exception error at runtime

\*\*\*\*\*Set 3\*\*\*\*\*

561) Qn 01:

```
public class TestDemo{
 public static void main(String args[]){
 LinkedList numList = new LinkedList();
 numList.add(1);
 numList.add(new Integer(2));//Line 1
 numList.add((int)2.5);//Line 2
 numList.add(new Integer('A'));//Line 3
 for(Integer num:numList){
 System.out.println(num);
 }
 }
}
```

ANSWER --> 1

2

2

65

=====

562) Qn 03:

```
public class Main{
 public static void main(String... sss){
 HashSet hashSet = new HashSet();
 String str1 = new String("Jack");
 String str2 = new String("Thomas");
 NameBean nameBean1 = new NameBean("Amold");
 NameBean nameBean2 = new NameBean("Diana");
 hashSet.add(str1);
 hashSet.add(str2);
 hashSet.add(nameBean1);
 hashSet.add(nameBean2);
 System.out.println(hashSet);
 }
}
```

ANSWER --> Compilation fails

=====

563) Qn 05:

```
int[] myArray = new int[]{1,2,3,4,5};
```

\*\*\*\*\*

=====

564) Qn 02:

```
public class SetImpl{
 public static void main(String[] args){
 List list=new ArrayList();
 list.add("Infosys");
 list.add("Google");
 list.add("IBM");
 for(String s:list){
 System.out.print(""+s);
 list.clear();
 }
 }
}
```

ANSWER --> An exception occurs at runtime

---

=====

565)

Qn 04:

```
public class TestDemo{
 public static void MyAppend(List iList){//Line1
 iList.add(007);
 }
 public static void main(String[] args){
 List iList=new ArrayList();//Line2
 MyAppend(iList);//Line3
 System.out.println(iList.get(0));
 }
}
```

```
}
```

ANSWER --> 7

```
=====
```

```
566) class Demo {
public static Collection get() {
 Collection s = new LinkedList();
 s.add("b");
 s.add("c");
 s.add("a");
 return s;
}
public static void main(String[]args){
 for(Object o:get()){
 System.out.println(o+" ");
 }
}
```

answer b c a

```
=====
```

```
567) class Demo {

public static void main(String[]args){
 HashMap p = new HashMap<>();
 p.put("key45", "somevalue");
 p.put("key12", "some other value");
 p.put("key39", "yet another value");
 Set s = p.keySet();
 s = new TreeSet(s); // line 1
 System.out.println(s);
```

```
}
```

```
answer
```

```
s = new TreeSet(s);
```

```
*****Set
1*****
```

```
568) Qn 35:(#Rep-2)
```

```
interface Interface1{
```

```
 default void method1(){
```

```
 System.out.println("Inside default method");
```

```
}
```

```
interface DefaultExtend extends Interface1{
```

```
 default void method1(){
```

```
 System.out.println("Default method redefined");
```

```
}
```

```
public class InterfaceWithDefaultMethod implements DefaultExtend{
```

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

```
 InterfaceWithDefaultMethod defaultExtend = new InterfaceWithDefaultMethod(); //Line4
```

```
 defaultExtend.method1(); //Line5
```

```
}
```

```
}
```

```
ANSWER --> Default method redefined
```

---

```
=====
```

  

```
569) Qn 36:
```

What happens if "default" keyword is omitted while defining a default method in interfaces?

```
interface Interface1{
 void method1(){
 System.out.println("Inside default method");
 }
}
```

method cant be overriden in the imp classes

method can be "

method cant be given body in the interface

compilation error

ANSWER --> b and c

---

570) Qn 37:(#Rep)

```
public class RepeatingAnnotations{
 @Retention(RetentionPolicy.RUNTIME)
 public @interface Chocolates{
 Favourite[] value() default{};
 }
 @Favourite("Dairy Milk")
 @Favourite("Kitkat")
 @Favourite("5 Star")
 @Favourite("Galaxy")
 public interface Chocolate{
 }
 @Repeatable(value = Chocolate.class)
```

```

public @interface Favourite{
String value();
}

public static void main(String args[]){
Favourite[] a = Chocolate.class.getAnnotationsByType(Favourite.class);
Chocolates Chocolate = Chocolate.class.getAnnotation(Chocolates.class); //Line5
for(Favourite favourite:chocolates.value()){
System.out.println(favourite.value());
}}}

```

Options - Nothing will display

- null will be printed
  - Runtime exception at line 5
  - Dairy Milk
  - Kitkat
  - 5 star
  - Galaxy
- 

571)

Qn 38:(#Rep-3)

```

public class TestDemo
{
public static void main(String args[]){
LocalDateTime date1 = LocalDateTime .of(2017,Month.FEBRUARY,11,15,30); //Line1
LocalDateTime date2 = LocalDateTime .of(2017,2,12,10,20);
System.out.println(date1.compareTo(date2));
}
}

```

ANSWER --> " -1 will be printed"

=====

572)

Qn 39:

```
public class DateDemo{
 public static void main(String args[]){
 String date = LocalDate.parse("2016-05-12").format(DateTimeFormatter.ISO_DATE_TIME);
 System.out.println(date);
 }
}
```

ANSWER --> Exception in thread "main" java.time.temporal.UnsupportedTemporalTypeException

\*\*\*\*\*Set  
2\*\*\*\*\*

573) Qn 03:(#Rep-2)

```
public class Formatting{
 public static void main(String args[]){
 LocalDate date = LocalDate.of(2016,11,13);
 DateTimeFormatter formatter = DateTimeFormatter.ofPattern("dd/MMM/yyyy", Locale.UK);
 System.out.println(date.format(formatter));
 }
}
```

ANSWER --> 13/Nov/2016

=====

574) Qn 04:(#Rep)

Is it possible to implement 2 diff. interfaces in a class having the same default method signature?

```
interface Interface1{
 default void method1(){
 System.out.println("Inside default method");
 }
}

interface DefaultExtend{
 default void method1(){
 System.out.println("Default method redefined");
 }
}

class Demo implements Interface1, DefaultExtend{
 public static void main(String args[]){
 Demo demo = new Demo();
 demo.method1(); //Line1
 }
}
```

ANSWER -->Not possible in Java 8

=====

575) Qn 05:

```
public class TestDemo{
 public static void main(String[] args){
 LocalDate date = LocalDate.of(12,11,2017);
 System.out.println(date);
 }
}
```

```
}
```

ANSWER --> Execution error as date component is not in range

```
*****Set
3*****
```

576) Qn 03:(#Rep)

```
interface DefaultMethodInterface1{
 default public void defaultMethod(){
 System.out.println("DefaultMethodInterface1");
 }
}
```

```
interface DefaultMethodInterface2{
 default public void defaultMethod(){
 System.out.println("DefaultMethodInterface2");
 }
}
```

```
public class TestDemo implements DefaultMethodInterface1,DefaultMethodInterface2{
 public static void main(String args[]){
 DefaultMethodInterface1 defMethIn = new TestDemo();
 defMethIn.defaultMethod();
 }
}
```

ANSWER --> Compilation fails

\*\*\*\*\*Set  
4\*\*\*\*\*

577) Qn 01:

```
class Apple
{
 A obj;
 Apple(A obj){this.obj=obj;}
 public A getObject(){return thi.obj;}
}

class Main
{
 public static void main(String[] args)
 {
 //Line1
 }
}
```

Output should be

76

Hello

Checkbox options for line 1

-Apple apple=new Apple(76);  
System.out.println(apple.getObject());  
Apple appleObj=new Apple ("Hello");  
System.out.println(appleObj.getObject());  
  
-Apple apple=new Apple(76);  
System.out.println(apple.getObject());  
Apple appleObj=new Apple("Hello");

```
System.out.println(appleObj.getObject());
```

```
-Apple apple=new Apple(76);
System.out.println(apple.getObject().toString());
Apple appleObj=new Apple("Hello");
System.out.println(appleObj.toString());
```

```
-Apple apple=new Apple(76);
System.out.println(apple.getObject().toString());
Apple appleObj;
appleObj=apple;
System.out.println(appleObj.toString());
```

---

```
*****Set
1*****
```

578) Qn 40:

```
String[] customers = new String[]{"John","Jack","Jacklin","Jasmine"};
List customersList = Arrays.asList(customers);
```

Best Code to iterate through the customersList obj data

i. for(Iterator itr = customersList.iterator();itr.hasNext();){  
System.out.println(itr.next());  
}

ii. for(String s:customerList)

```
System.out.println(s);
```

```
iii. for(Iterator itr = customersList.iterator();itr.next();){
 System.out.println(itr.next());
}
```

Option --> i only

```
=====
```

```
==
```

579)

Qn 41:

Best exception handling practices??(Checkboxes)

1) public void display() throws FileNotFoundException, SecurityException

2) public void display() throws Exception

3) catch(FileNotFoundException e){

```
 throw new CustomerException("Error occurred"+e)
}
```

4) catch(FileNotFoundException e){

```
 logger.error("Error Occurred",e);
 return null;
}
```

5) catch(FileNotFoundException e){

```
e.printStackTrace();

return null;

}
```

ANSWER --> 1,4,5

=====

=====

580)

Qn 42:(#Rep)

What is magic number in terms of Java pgmming best practices

ANSWER --> Direct usage of the number in the code

=====

=====

581) Qn 43:(#Rep)

1. public class Ex1{
2. public String formatInput(String i){
3. if(i.trim().length()==9){ // Compilation error?
4. StringBuilder s1 = new StringBuilder();
5. s1 = s1.insert(0,"+1(");
6. s1 = s1.insert(6,")"); // Compilation error?
7. s1 = s1.insert(10,"-"); // Compilation error?
8. return s1.toString();
9. }
10. return null;
11. }
12. public static void main(String[] args){

```
13. Ex1 ob = new Ex1();
14. String i;
15. ob.formatInput(i); //Null pointer Exception if i is null
16. } }
```

ANSWER --> //Null pointer Exception if i is null

=====

582) Qn 44:(#Rep-3)

Which is valid?

```
1. public void draw(String s){
 if(s.equals("Square")){
 drawSquare();
 }
 if(s.equals("Rectangle")){
 drawRectangle();
 } }
```

```
2. public void draw(String s){
 if("Square".equals(s)){
 drawSquare();
 }
 if("Rectangle".equals(s)){
 drawRectangle();
 } }
```

ANSWER -->

option 1

\*\*\*\*\*Set  
2\*\*\*\*\*

583) Qn 03:

```
public class TestDemo{
 public static void main(String args[]){
 int i = 4;
 int j = 4;
 System.out.println(i==j);
 Integer w1 = new Integer(4);
 Integer w2 = new Integer(4);
 System.out.println(w1==w2);
 }
}
```

what is the issue with the above code?

- a) Comparison of wrapper using == is wrong
- b) Comparison of primitive data types using == is wrong
- c) both a and b
- d) no issues

ANSWER --> No issues

---

584) Qn 04:(#Rep)

```

public class Demo{
 private static String id;
 private Random random = new Random();
 public Demo(){
 //Line 5
 id = "ACC1101"+Math.abs(random.nextInt());
 }
}

public String getId(){
 Return id;
}

```

which to be inserted at Line 5 considering the code is running in a multi threaded environment?

- a) if(id==null)
- b) synchronized(this){if(id==null){
- c) if(id==null){synchronized(this){
- d) synchronized{if(id==null){

=====

585)

Qn 05:

which of the foll. is the best defensive code snippets?

- a) Date dobj1 = new Date();  
 Employee e1 = new Employee(d1);  
 dobj.setYear(1988);

```
b) Class Employee{
 private final Date doj;

 Employee(Date date){
 this.doj=new Date(date.getTime());
 }

 public Date getDateOfJoining(){
 return this.doj;
 }
}
```

```
c) Date dobj1 = new Date();

Employee e1 = new Employee(d1);

Date dobj2 = e1.getDateOfJoining();

dobj2.setYear(2015);
```

```
d) Class Employee{
 private final Date doj;

 Employee(Date date){
 this.doj=new Date(date.getTime());
 }

 public Date getDateOfJoining(){
 return new Date(this.dateOfJoining.getTime());
 }
}
```

ANSWER - b

```
*****Set
3*****
```

ALL REPEATED

```
*****Set
4*****
```

586) Qn 01:

```
public class TestDemo{
 private static Object staticObject;
 public static Object createStaticObject(){
 if(staticObject==null){
 staticObject=new Object();
 }
 return staticObject;
 }
}
```

Changes required for successful execution

- the method createStaticObject should be synchronized
- the method createStaticObject should be private
- the staticObject reference should not be static
- The method createStaticObject should not return Object type

```
=====
```

587) Qn 02:

```
public class Employee{
```

```
private int empld;
private String empName;
private String designation;
private transient String dob;
}
```

choose 2

- Fields in non-serializable classes should not be "transient"
- Fields in non-serializable classes should be "transient"
- Fields in non-serializable classes should either be "transient" or serializable
- Make the Employee class serializable

=====

588) Qn 03:

```
public class Ex1{
 public static void main(String args[]){
 int i=34;
 int j=34;
 System.out.println(i==j);
 Integer i1=new Integer(34);
 Integer i2=new Integer(34);
 System.out.println(i1==i2);
 }
}
```

ANSWER --> true

false

=====

=

589) Qn 04:

Which of the naming convention is valid for method names?

ANSWER --> methodName

\*\*\*\*\*Set  
1\*\*\*\*\*

590) Qn 45:

- i) Gather the information for an object gradually before requesting its construction -- a) Prototype
- ii) Defer the decision of which class to instantiate -- b) Abstract Factory
- iii) Construct a family of objects that share some trait -- c) Builder
- iv) Specify an object to create by giving the example -- d) Factory Method

ANSWER -->

- i) -> c ii) -> d iii)-> b iv) -> a

=====

=====

591) Qn 46:

Which design pattern can be used to return factories which can be used to create set of related objects?

ANSWER --> Abstract Method

=====

=====

592) Qn 47:

How to make a class to follow Singleton pattern?

- a) Singleton obj can be replaced with encapsulated setter method
- b) Requires constructor of singleton class to be private
- c) Singleton obj must be named instance
- d) Ensures that there is only one insatnce of an object in memo
- e) Requires a public static method to retrieve the instance of the singleton

ANSWER --> b, d, e(Not sure about c)

\*\*\*\*\*Set  
2\*\*\*\*\*

593) Qn 01:

what changes need to be done to make the below code to follow a singleton pattern?

```
public class Employee{
 public static Employee C;
 private Employee(){}
 public static Employee getEmployee()
 {
 if(employeeInstance==null){
 employeeInstance = new Employee();
 }
 }
```

```
 return employeeInstance;
}
}
```

options:

1. None
2. Rename Employee to instance

```
=====
=====
```

594)

Qn 02:

```
interface Prototype{public Prototype doClone();}

class Person implements Prototype{

 String name;

 public Person(String name) {this.name = name;}

 @Override

 public Prototype doClone(){return new Person(name);}
}
```

```
public class Demo{

 public static void main(String[] args){

 Person person1 = new Person("Clone");

 Person person2 = (Person) person1.doClone();

 Person person3 = person1;

 System.out.println((person1==person2)+" "+(person1==person3));
 }
}
```

ANSWER --> false true

=====

=====

595) Qn 03:

How to make the class follow Singleton pattern?

- a) Singleton obj can be replaced with encapsulated setter method
- b) Requires constructor of singleton class to be private =====> correct
- c) Singleton obj must be named instance
- d) Ensures that there is only one instance of an object in memory =====> correct
- e) Requires a public static method to retrieve the instance of the singleton =====> correct

\*\*\*\*\*Set

3\*\*\*\*\*

Qn 01:

which are true?

1. java.util.Calendar class imp factory design pattern
2. java.util.ResourceBundle.getBundle() imp factory design pattern
3. java.text.NumberFormat.getInstance imp Singleton pattern design pattern
4. java.awt.Desktop.getDesktop() imp singleton pattern design pattern
5. java.lang.Runtime.getRuntime() imp Abstract Factory design pattern

a) 1,2,4

b) 1,2,3,4,5

c) 4,5,2

d) 1,3,4

e) 2,3,5

ANSWER --> 1,2,4

=====

=====

596) Qn 02:

```
Locale locale = new Locale("da","DK");
NumberFormat numberFormat = NumberFormat.getInstance(locale);
String number = numberFormat.format(100.99);
System.out.println(number);
```

NumberFormat.getInstance() follows which pattern?

- a) Factory Method pattern
- b) Singleton pattern
- c) Abstract Factory pattern
- d) Builder pattern

ANSWER --> Factory Method pattern

=====

=====

597) Qn 03:

which indicates the need to use factory pattern?

- a) We have 2 classes that do the same thing
- b) We only want one instance of the object to exist
- c) We want to build a chain of objects
- d) We don't want the caller to depend on a specific implementation

ANSWER --> d

\*\*\*\*\*SET  
4\*\*\*\*\*  
all repeated

Qn 48:

```
@RunWith(Suite.class)
@Suite.SuiteClasses({
 Sample.class,
 TestClass.class})
private class JunitTestSuite{}
```

Ans - Compilation error

=====

598) Qn 49:

```
public class TestClass
{@Test
public void test(){
 String a = "";
 assertEquals("a",a);
}}
```

Ans - Test fails

=====

599) Qn 50:(#Rep)

Loopdemo.java

```
public class Loopdemo{
 public char display(int marks){
 if(marks>=85&&marks<99){
 return 'A';}
 else if(marks>65 && marks<=84){
 return 'B';}
 else{
 return 'F';}}}
```

TestLoop.java

```
public class Testloop(){
 value = tc.display(80);
 Assert.assertSame(value,'B');
}
```

Options:

Out of branches ; Coverage

8 - 3

7 - 2

5 - 1

5 - 2

=====

600) Qn 01:

```
public class TestDemo{
 @Test
 public void test(){
 fail("Not yet implemented");//Line 1
 assertEquals(1,2);//Line2
 }
}
```

ANSWER --> error is in both line

=====

601) Qn 02:(#Rep)

```
@RunWith(Suite.class)
@Suite.SuiteClasses({
 JunitTestSuite.class,
 MyNewTest.class})
public class JunitTestSuite1{}
```

How many tests will run ?

- a) 1
- b) 2
- c) 3
- d) Compilation error

Ans-b

=====

602) Qn 03:

```
public class MyNewTest{
 @Test
 public void test1(){
 //
 }
}
```

JUnitTestSuite.java is a Test Suite with 2 classes >> FirstTestClass , SecondTestClass

```
@RunWith(Suite.class)
@Ignore
@Suite.SuiteClasses({
 FirstTestClass.class,
 SecondTestClass.class
})
public class JunitTestSuite{
}
```

JunitTestSuite1.java is a Test Suite defined as follows

```
@RunWith(Suite.class)
@Suite.SuiteClasses({
 JUnitTestSuite.class,
 MyNewTest.class
```

```
})
public class JunitTestSuite1{
}
Ans -2
```

=====

603) Qn 01: --> Solved

```
class MyTest{
protected int display(int a, int b){
return a+b;
}
}
public class TestDemo extends MyTest{
@Test
public void test1{
assertEquals(3,display(1,2));
}
}
```

- a) Run time exception
- b) Test cases fail
- c) Test cases pass
- d) Compilation fails

Ans-c

=====

604) Qn 03: --> Solved

```
public class C1{
 @Test
 public void testEqual(){
 Assert.assertEquals(2,2);
 }

 public class C2{
 @Test
 public void testEqual(){
 Assert.assertEquals(2,2);
 }
 }
 @RunWith(Suite.class)
 @SuiteClasses({C1.class,C2.class})
 public class TestDemo{
 @BeforeClass
 public class TestDemo{
 @BeforeClass
 public static void setUpClass(){
 System.out.println("Master setup");
 }
 @Before
 public void setUp{
 System.out.println("Slave setup");
 }
 @Test
 public void test(){
 assertEquals(2,2);
 }
 }
 }
}
```

```
}
```

```
@AfterClass
```

```
public static void tearDownClass(){
```

```
System.out.println("Master tearDown");
```

```
}
```

```
@After
```

```
public static void tearDown(){
```

```
System.out.println("Slave tearDown");
```

```
}
```

```
}
```

Predict the output...

a) 3 TCs run and in console the o/p is displayed as

Master setup

Master tearDown

b) 2 TCs run and in console the o/p is

Master setup

Master tearDown

c) 3 TCs run and in console the o/p is displayed as

Master setup

Ans-b



1. What is true with respect to the abstract class being given below?

605) Abstract class Employee

```
{
//fields and constructor
Public void salaryCompute()
{
//code goes here
}
```

```

Public abstract void taxReduce() -- If method is defined as abstract the last character in the
line should be ";"

{

//code goes here

}

Public abstract void benefitsInclude();

}

- Class Employee should be private as it is abstract
- Class Employee should be public as it is abstract
- Class Employee cannot be abstract as it has concrete methods/method definitions
- Abstract method of class Employee has definition -- The method taxReduce() has
definition

```

606) Annie and Jacklin are working on a Java project. Annie is working on a Windows machine whereas Jacklin is working on a Mac machine. Which feature of Java helps Annie and Jacklin's projects to execute on each other's machines, even though they are working on different components?

- Multithreading
- Object oriented
- Architecture neutral and portable
- Memory management

607) What is the output when the below code is executed?

```

Public class Demo{
 Public static void main(String args[]){
 For (int i = 0;i<5;i++){
 Switch(i){
 Case 0: System.out("v ");break; -- Actually this will give compilation error but if
System.out is changed to System.out.print the program will execute
 Case 1: System.out("w ");
 Case 2: System.out("x ");
 Case 3: System.out("y ");
 Case 4: System.out("z ");break;
 }
 }
 }
}

- v w x y z y z z
- v w x y z x y z y z z
- v w x x y z z
- v w x y z

```

608) Ria has a class called 'Account.java' under two separate packages com.infy.debit and com.infy.credit. Can she use the Account class of both packages in another class called 'ReportUtil.java'l of package com.infy.util? **Can use Fully qualified className instead**

- Yes, she can use
- No, she cannot as there will be a compilation error stating the import collides with another import
- No, she cannot. The code will pass compilation but an ambiguity will get encountered during the execution.
- No, she cannot as there will be a compilation error while creating the Account class for the second time though in a different package

609) Class Expression

```
{
Public void calc()
{
 Double x = 10;
 Int y = 20;
 Float z = 30;
 //line 1
 b = x+y+z; -- It's a combination of all 3 so higher Data Type will not have any data loss.
}
}
```

Identify the suitable datatype to be used for variable "b" at Line 1?

- Long
- Int
- Double
- Float

610) Which of the below 'if' statement can be used to find a year is a leap year or not?

- If(((year % 4 == 4) && (year % 100! = 0)) || (year%400 ==4 ))
- If(((year % 4 == 4) && (year % 100! = 0)) || (year%400 ==0 ))
- If(((year % 4 == 4) && (year % 100! = 4)) || (year%400 ==4 ))
- If(((year % 4 == 0) && (year % 100! = 0)) || (year%400 ==0 )) -- Year divided by 4 should return mod as 0, year divided by 100 should not be 100 because year is 4 digits, year divided by 400 should be 0

611) Which of the following class definitions belong to an abstract class?

- Class A ( abstract void unfinished() {})
- Class A (abstract void unfinished();)
- Abstract class A { abstract void unfinished();}
- Public class abstract A { abstract void unfinished();} -- abstract keyword is misplaced

612) Class Child extends Parent{ -- No definition available for Parent class ,that will cause compilation error

613) Int display1(int i)//Line 1

```
{
```

2.       Return display2(++i); //Line2 -- No definition available for display2 method, that will cause compilation error

```
}
```

```
}
```

```
Public class Test {
```

```
 Public static void main(String[] args){
```

```
 System.out.println("Value is " + new Child().display1(564));
```

```
}
```

```
}
```

- Value is 565
- Value is 566
- Value is 567
- Value is 568

- 614) Which of the following feature comes under compile time polymorphism in Java? Choose any two )

- Method overloading --
- Constructor overloading -- overloading is usually compileTime
- Method overriding --
- Constructor overriding -- Overriding is Runtime

- 615) Predict the output of the below code

```
Class Book{
```

```
 Int bookid = 2356;
```

```
}
```

```
Class Book1 extends Book{
```

```
 Int bookId = 1167;
```

```
}
```

```
Class Book2 extends Book1{
```

```
 Int bookId = 2378; //Line 8
```

```
 Void display(){
```

```
 System.out.print(super.super.bookId); //Line 10 --compilation error as "super." is
```

repeated twice. May be typo. If "super." Is added only once then no compilation error.

```
 System.out.print(super.bookId); //Line 11
```

```
 System.out.print(bookId);
```

```
}
```

```
}
```

```
Class Demo{
```

```
 Public static void main(String arg[]){
```

```
 Book2 book2 = new Book2();
```

```
 Book2.display(); --- compilation error here as display() method is called using Class
```

Name and display is not static. If this line is "book2.display(); " instead , no compilation error.

```
}
```

- }
- Compilation fails because of an error in Line 10 as “super” keyword is unexpected
  - Compilation fails because of an error in Line 11 as variable “bookId” keyword is not defined in parent class as public
  - Compilation fails because of an error in Line 8 as variable “bookId” cannot be defined in child class
  - Code runs and gives output 2356 1167 2378 - No option is correct
- If Line 10 is “`System.out.print(super.bookId);`” then output will be 116711672378,  
 else if Line 10 is “`System.out.print(bookId);`” then output will be  
 237811672378, If Line 10 is removed and placed after “Class Book1 extends Book{}”  
 line then output will be 2356 1167 2378

616) Public static void main(String[] args) throws Exception{

```
Try{
 System.out.print("Greetings!!" + " " + args[0]); -- args[0] will throw
 "ArrayIndexOutOfBoundsException" Exception . So this line will not be executed . Only
 the catch block will be executed.
```

```
} catch (ArrayIndexOutOfBoundsException e) {
 System.out.print("Sam");
}
```

```
}
```

```
}
```

Predict the output?

- Greetings!! Sam
- Greetings!!
- Sam
- Greetings!! args[0]

617) What is the output of the following code when executed?

```
Public class StringTester{
 Public static void main(String[] args){
 String name = new String("Jordan");
 StringBuffer newname = new StringBuffer(name);
 System.out.println(name.equals(newname)); -- StringBuffer returns true only
when compared to itself not with any other StringBuffer
 "System.out.println(newname.equals(newname)); " will return true
 }
}
```

- False
- True
- NullPointerException
- Compilation Error

618) Public class StringEquals{

```

Public static void main(String[] args){
 String name1 = "Infosys";
 String firstName = "Inf";
 String lastName = "osys";
 String name2 = firstName.concat(lastName);
 System.out.print(name1.equals(name2)); -- equals compares String value
 System.out.print(name1==name2); -- == compares object address(reference)
}

```

- False false
- True false
- True true
- False true

619) What is the output of the code snippet the below code?

```

Public class StrintTest{
 Public static void main("String[] args){
 String s1 = "JAVA", s2 = "java";
 S1.toLowerCase(); -- The updated value is not retained
 System.out.print((s1.equals(s2))); -- Here original s1 with "JAVA" is compared so false
 }
}

```

- False
- True

620) Public class Tester{

```

 Public static void main(String[] args){
 Set<Integer> set1 = new Hashmap<>(new Comparator<Integer>(){ -- Map in
Java should contain 2 values, here only 1 is provided and that to a type of comparator
which cannot be added to Map
 @Override
 Public int compare(Integer o1, Integer o2){
 Return o2.compareTo(o1);
 });
 set1.add(234);
 set1.add(657);
 set1.add(143);
 System.out.println(set1);
 }
}

```

- [234,657,143]
- [143,234,657]
- Compilation Error: Cannot infer Type argument for HashMap.
- [657,234,143]

621) Which of the following interfaces are not a part of Java Collection framework?(Choose any 2)

- List
- Queue
- SortedList
- ArrayList

622) What will be the output of the following code when executed?

```
Import java.time.LocalDate;
```

```
Public class LocalDateTester {
 Public static void main(String[] args){
 LocalDate local = LocalDate.of(2020,3,20);
 Local = local.minusWeeks(-4L); -- minus and (-)will become + 4 weeks, So
 output would be 2020-04-17
 System.out.println(local);
 }
}
```

- 2020-02-29
- 2020-04-28
- 2020-04-27
- 2020-02-28

623) Assume we have declared a static method with same name in the two interfaces and a class tries to implement both the interface result?

- The code will not be complied due to Static method of an interface cannot be overridden
- The code will not be complied as two interface has same name for the static method
- The code will not be complied we have to override the static method in the class to avoid the Diamond problem
- The code will compile successfully

624) What will be the output of the following code when executed?

```
Public class DateTimeTester{
 Public static void main(String[] args){
 LocalDateTime localDateTime = LocalDateTime.of(2020,5,13,20,46);

 System.out.println(localDateTime.get(ChronoField.HOUR_OF_DAY)+localDateTime.getDayOfMonth()); -- ChronoField.HOUR_OF_DAY will return a int value 20 ,
 localDateTime.getDayOfMonth() will return a int value 13 , Due to "+" symbol inbetween 2
 integers , both values will get added and output will be 33.
 }
}
```

- 13
- 2013
- 33

o 5

625) From the below options identify the methods and constructions in Throwable that support chained exceptions

- i. Throwable getCause()
- ii. Throwable intiCause(Throwable) -- Incorrect Spelling, Actually Throwable initCause(Throwable) supports chained Exception
- iii. Throwable(String, Throwable)
- iv. Throwable(Throwable)
  - o Options iii and iv only
  - o Options i,ii,iii,iv
  - o Option iv only
  - o Options i and iii, and iv only

626) Which of the below is not a valid classification of design pattern?

- o Creational patterns
- o Structural patterns
- o Behavioural patterns
- o Java patterns

627) Which of the following annotations are used for creating test categories?

- o @Category and @RunWith
- o @Category and @Suite
- o @Suite and @RunWith
- o @Categorize and @RunWith

628) Given

```
Public class App{
 Public void display() throws IOException{
 Throw new IOException();
 }
}

Public class AppTest{
 Public void testDisplay() throws IOException{
 App obj = new App();
 obj.display();
 }
}
```

Select the option that can be used in Line1 for the TestCase to fail?(choose 2)

- o @Test(expected = IOException.class)
- o @Test(expected = FileNotFoundException.class)

- @Test(expected = Exception.class)
- @Test

629) Given:

```
//Assume all the required imports are added
Loopdemo.java
Public class LoopDemo{
Public class display(int marks){
 If (marks>=85&&marks<99){
 Return 'A'; }
 Else if (marks>65&& marks<=84){
 Return 'B'; }
 Else{
 Return 'F'; } }

TestLoop.java
public class Testloop{
 Loopdemo tc=new Loopdemo();
 Char value;

@Test
Public void testdisplay(){
 Assert.assertSame(tc.display(70),'B');
//Line 1
}
}

To get 100% branch coverage which code should be added in Line 1?
```

- **Assert.assertSame(tc.display(88),'A');**  
**Assert.assertSame(tc.display(64),'F');**  
**Assert.assertNotSame(tc.display(100),'F');**
- **Assert.assertSame(tc.display(88),'A');**  
**Assert.assertSame(tc.display(64),'F');**
- **Assert.assertSame(tc.display(85),'A');**  
**Assert.assertSame(tc.display(64),'F');**  
**Assert.assertSame(tc.display(84),'B');**
- **Assert.assertSame(tc.display(85),'B');**  
**Assert.assertSame(tc.display(64),'F');**  
**Assert.assertNotSame(tc.display(74),'A');**

630) The below code will generate compilation error. Select the possible options to avoid it. (Choose )

```
Public class App {
 Public static void main(String[] args){
 String msg = null;
 Try{
```

```

 System.out.println(msg.length());
 } catch (NullPointerException ex){
 System.out.println("Exception is caught here");
 Throw ex; //Line 1
 System.out.println(msg); //Line2
 }
}

- Place Line2 before Line1
- Remove Line2 i.e. After throw there should not be any statements
- In Line2 change the msg as System.out.println(null);
- Replace Line1 with ex.printStackTrace();

```

631) Public class OperatorsDemo {

```

Public static void main(String[] args)
{
Int x = 120, y =110;
String s1 = "Thank", s2 = "Thank";
Int arr1[] = {1,2,3};
Int arr2[] = {1,2,3};
Boolean boo = true;
1.System.out.println("x==y:"+(x==y)); 120 is not equal to 110 . So, output is "false".
2.System.out.println("(x<=y):"+(x<=y)); 120 is not lesser than 110 . So, output is "false".
3.System.out.println("s1==s2:"+(arr1==arr2)); arr1 & arr2 are objects so cannot be compared
with "==" . So, output is "false"
4. System.out.println("boo==true:"+(boo==true)); value of "boo" is true . So, output is "true"
}
}

- x == y:false
x<=y:false
s1==s2:true
boo==true:false
- x==y:false
(x<=y):false
s1==s2:true
boo==true:true
- x==y:true
x<yfalse
s1==s2:false
boo==true:true

```

- x==y:false
- (x<=y):false
- s1==s2:false
- boo==true:true

632) Which of the following is the correct usage of a relationship operator inside an if statement?

- If(firstName == "Annie")
- If(firstName.equals("Annie"))
- If(firstName.equals("Annie") && salary == 50000)
- If(firstName.equals("Annie") | !salary == 50000) -- Not operator(!) is misplaced . The right ways is If(firstName.equals("Annie") | salary != 50000)

633) Identify the output

```
Public class MyDemo {
 Public static void main(String[] args) {
 Int i = 5;
 Switch(i) {
 Case 1:
 System.out.println("One");
 Break;
 Case 2: // Line 1
 Case 3: // Line 2
 System.out.println("Two and Three");
 Case 4,5: //Line 3
 System.out.println("Four and five");
 Break;
 Default:
 System.out.println("Default");
 }
 }
}
```

- Compilation error in Line 1 as there is no body for this case
- Compilation error in Line 2 as there is no break statement.

Compilation error in Line 3 as multiple values are not allowed in case -- case 4: case 5: //Line 3 is correct

- It will work fine and display "Four and Five" as output.

634) Given

```
Class Invoice {
 Int count = 100;
```

```

Void billNo() throws Exception {
Try {
 Count++; value of count is increased to 101
 throw new Exception(); Exception is thrown
} catch (Exception ex) {
 Count++; value of count is increased to 102
}
}
Void display() {
System.out.println("Bill no.: " + count); printing the value of count
}
Public static void main(String[] args) throws Exception {
Invoice inv = new Invoice();
Int.billNo();
Inv.display();
}
}
Predict the output?

- Bill no.: 103
- Bill no.: 102
- Bill no.: 101
- Bill no.: 100

```

- 635) Have a look at the following class and predict the option that is correct.

```

Class CodeForException
{
Public void callMe() throws Exception
{
Try
{
Int value = 3/0;
}
Catch(ArithmeticException ae)
{
System.out.println(ae);
}
}
Public void calling()
{
callMe();
}
}

- The code will face issues during compilation as the calling code neither handles nor throws Exception

```

- The code seems to be perfect and will pass compilation
- The code will face issues during compilation as callMe() has code for handling exception and throws the Exception as well.
- The code will face issues during compilation as the unchecked exception ArithmeticException is handled using catch block

636) Class AccessModifier

```
{
 Public int y =20;
 Float z = 30;
 Protected int a =20;
 Private int b = 20;
}
```

Identify the order of the variables based on the visibility from high to low

- yazb**
- azby
- zaby
- abyz

637) The below code will generate compilation error. Select the possible options to avoid it. (Choose 3)

```
Public class App {
 Public static void main(String[] args){
 Try {
 System.out.println(msg.length());
 } catch (NullPointerException ex) {
 System.out.println("Exception is caught here");
 throw ex; // Line 1
 System.out.println(msg); // Line2
 }
 }
}
```

- Place Line2 before Line1
- Remove Line2 i.e. After throw there should not be any statements
- In Line2 change the msg as System.out.println(null);
- Replace Line1 with ex.printStackTrace();

638) Predict the output of the below code:

```
Class Car {
 Void start() {
 System.out.println("Car Starts");
 }
}
```

```

Class Bike{
Void start() {
System.out.println("Bike Starts");
}
}

Class Automobile extends Car {
Void start() {
System.out.println("Automobile Starts");
}
}

Public class ExceptionDemo{
Public static void main (String[] args){
System.out.println("Implementing type casting");
Car d = new Car(); An object of SuperType(car) is created
Automobile automobile = (Automobile) d; A super type(Car) object is assigned to sub
type(Automobile) object which will give classcast Exception
Automobile.start();
}
}

```

- Displays “Implementing type casting” in Console
- Displays “Implementing type casting” and RUNTIME EXCEPTION :  
java.lang.ClassCastException

**Downcasting Example for correct result:**

```

Automobile automobile = new Automobile(); -- create an object of SubType
Car d = (Car) automobile; -- SubType object is assigned to SuperType Object
d.start();

```

- Displays “RUNTIME EXCEPTION : java.lang. ClassCastException”
- Displays “Automobile Starts” in Console

639) Which of the following condition will not allow the finally block be executed?

- When some error occurs
- When exception is raised
  - When system.exit(1) is called -- Program will terminate abruptly so finally will not be called.
- When exception is not raised

640) While Jacob was writing the below code, he came across a compilation error ? Please identify and help Jacob.

```

Public class TestDemo {
Public static void main(String[] args) {
Try {//line 1
 System.out.pirnt("In try");
 Return; //line 2
} finally { //line 3

```

```

 System.out.print(" In finally");
 }
 System.out.print("Outside block"); //line 4
}
}

- At line 1 because try block can't be written in main() method.
- At line 2 because no value is returned from 'return' statement.
- At line 3 because finally block is not allowed after a try block
- At line 4 because of unreachable code. – finally is the last block , anything after finally will not be reachable

```

641) Which of the following option can be inserted in line5 to get the output as "SAMsung"?

1. Public class Util {
2. Public static void main(String[] args){
3. String s = "SAM";
4. S.toLowerCase();
5. //insert code here
6. System.out.println(s);
7. }
8. }
  - S.replace("SAM","SAMsung");
  - S.join("sung"); -Join method accepts 2 parameters not 1
  - S.concat("sung");
  - None of the option is correct. String in Java is immutable So whenever we replace or concat a new String reference should be created . The below code will work
 

```
String s1 = s.replace("SAM", "SAMsung");
System.out.println(s1);
(Or)
String s1 = s.concat("sung");
System.out.println(s1);
```

642) Which among the following code snippets are illustrating the concept of **autoboxing** here?  
(Choose any 2)

- Character ch = 'a'
- ArrayList<Integer>listOfTickets = new ArrayList<>();
 

```
listOfTickets.add(101);
listOfTickets.add(102);
```
- Int var1 = new Integer(1003); --This is Boxing and not Auto-Boxing
- ArrayList arrayList = new ArrayList();
 

```
Int number = arrayList.get(0); -- Type mismatch
```

643) What is the result when the following code is compiled and executed?

```

Class Employee<E,A>{ – Generics notification in class definition
 E eObj1; -- Generics notified in class definition is declared in it's own type(class)
}

```

A aObj1; -- Generics notified in class definition is declared in it's own type(class)

```
Employee(E eObj1, A aObj1){
 This.eObj1 = eObj1; -- Values of the Generic variable are received while instantiated and
 is set to the class
 This.aObj1 = aObj1;
}
Public void display(){
 System.out.println(eObj1);
 System.out.println(aObj1);
}
}
Public class Demo{
 Public static void main(String[] args){
 Employee<String, Integer> employee = new Employee<>("Annie Thomas",25); -- The
 Values are sent while instantiating the object and facilitates the constructor to receive and
 attach to the class
 employee.display();
 }
}
}

- Null
- Null
- 25
- Annie Thomas
- 25
- Annie Thomas
- 0

```

644) Which among the following is valid option for wildcards?(Select 2 options)

- Used to relax restriction on the variable -- UpperBound wildcard
- Used in scenario where type being operated upon is not known -- Unbound wildcard
- Used in generic method type argument
- Can access members of super class

645) Public class Util{

```
Public static void main(String par[]){
 1. LocalDate date = LocalDate.of(2019, 3, 17);
 2. date = date.minusDays(18); - 2019 February had 28 days . So, minus 18 from march 17 is
 27th February
 3. date.minusMonths(1); -- 1 month is subtracted but is not saved in a variable or printed .
 4. System.out.Println(date); -- date value in line 2 is printed which is February 27
}
}
```

Predict the output?

- 2019-02-28
- 2019-02-27
- 2019-03-01
- 2019-03-02

646) Which of the following are the correct way to declare a generic method in JAVA? (Choose any 3)

- access-specifier <generic-type-parameter-list> return-type method-name(parameter-list){} -- default ,Non-static method
- access-specifier static-keyword <generic-type-parameter-list> return-type method-name(parameter-list){} -- static method
- access-specifier return-type <generic-type-parameter-list> method-name(parameter-list){}
- <generic-type-parameter-list> return-type method-name(parameter-list){} default non-static method

647) Which of the below statements are true about design patterns?

- 1) There are only three design patterns defined in JAVA
- 2) We can use each design pattern only once per application
- 3) Design patterns are conceptual reusable solutions
  - Statements 1,2,3
  - Statements 3,2
  - Only Statement 2
  - Only Statement 3

648) Which among the following is valid option for wildcards?(Select 2 options)

- Used to relax restriction on the variable –UpperBound Wildcard
- Used in scenario where type being operated upon is not known -- Unbound Wildcard
- Used in generic method type argument
- Can access members of super class

649) Which one of the following is used for the automatic accurate tracking for the decimal values?

- Float
- Double
- BigDecimal
- Decimal

650) Missed few lines

```
.....
}
}
```

```
Public class CarParking implements Parking{
```

```
Static void park(){ -- Implementation of park() method in CarParking is static ,So confirming the
park() method in Parking Interface as static
```

```
System.out.println("From CarParking class");
}
Public static void main(String[] args){
 Parking.park(); -- Since park() method is called using Interface name assuming the park()
method in Parking Interface as static
 park();
}
```

What will be the output of the code after execution?

- From Parking Interface From CarParking class -- Parking Interface definition is not available so selected this option based on above assumptions
- From Parking Interface From Parking Interface
- Compilation error as park() is undefined for class CarParking.
- Compilation error as cannot call park() with Parking reference.

651) import java.time.LocalDate;

```
public class DateTester {
 public static void main(String[] args) {
 LocalDate local=LocalDate.of(2020,1,1);
 local=local.minusMonths(-5);
 local=local.minusDays(9);
 System.out.println(local);
 }
}
```

- A. 2020-05-23
- B. 2019-07-23
- C. 2020-01-01
- D. 2020-06-23

1.Which of the jump statement can skip processing of one iteration if a specified condition occurs and process with the remaining iterations?

- a. break
- b. return
- c. continue
- d. exit

**Answer: continue**

2. which of the following class/interface is not part of the collection family

- a.Queue
- b.Enqueue
- c.Dequeue
- d.ArrayDeque

**Answer: Enqueue**

3. which among the following option are correct with respect to Hashmap(select 2 options)

- a.override boolean equals(object o)
- b.override to(String)
- c.override int hashCode()
- d.override String hash(code)

**Answer: override boolean equals(object o) & override int hashCode()**

4.which of the below method name is valid as per java naming convention?

- a.METHOD\_NAME
- b.MethodNAme
- c.methodName
- d.Method\_Name

**Answer: methodName**

5. which one of the following is not used for the automatic accurate tracking for the decimal values?

- a.float
- b.double
- c.BigDecimal
- d.Decimal

**Answer: BigDecimal**

6. what is the magic number in java in the concept of java programming best practices

- a.A number which gets printed on the console
- b.A direct usage of the number in the code
- c.A number which magically disappears from the code
- d.A number which is generated through error

**Answer: A direct usage of the number in the code**

7. which of the following mechanism in OOP is a process of hiding certain details and showing only essential information to the user?

- a.polymorphism
- b.encapsulation
- c.abstraction
- d.inheritance

**Answer: Abstraction**

8. which of the following is false regarding errors in java exception handling?

- a.An error shows a serious problem which should not be caught by an application
- b.An error usually occurs due to lack in any of the system resources
- c.Errors can be handled by the code of the program
- d.An error is taken as the unchecked exception

**Answer: Errors can be handled by the code of the program**

9.what is the output:

```
public class ABC {
 public static void main(String args[])
 boolean flag = false
 if(flag = true) {
 system.out.println("true");
 }
 else {
 system.out.println("false");
 }
}
```

- a.true
- b.false
- c.true false

**Answer: true**

10.which of the following comes under compile time polymorphism in java(choose any 2)

- a.method overloading
- b.Constructor overloading
- c.Method overriding
- d.constructor overriding

**Answer: Method Overloading & Constructor Overloading**

11.how many number of values can be accomodated by varargs in java?

- a.atleast one value
- b.atmost two values

- c.more than one value
- d.any number of values

**Answer: atleast one value**

12.which of the following is/are true about abstract class in java

- (i)object of an abstract class cant be created
  - (ii) an abstract class is designed only to act as a base class in hierarchy to be inherited by other class
- a.only(i)
  - b.only(ii)
  - c.both(i) and (ii)
  - d.neither(i) and (ii)

**Answer: both (i) and (ii)**

13.which of the following interface should be implemented by a resource if we want to create a resource part of try block

- a.closeable
- b.closeable<E>
- c.AutoCloseable
- d.AutoCloseable<E>

**Answer: AutoCloseable**

14.A hash code of an object can be defined as?

- a.unique code of an object
- b.unique ID of an object
- c.unique methods of an object
- d.unique representation of an object

**Answer: Unique ID of an object**

15. which of the following interface is implemented by ArrayList and linkedlist?

- a.collection
- b.Iterable
- c.List
- d.Map

**Answer: List**

16. What is the output?

```
public class GenericTester {
 public static void main(String args[]) {
 List<Integer> list = new ArrayList<>();
 list.add(22);
 list.add(23);
 method(list);
 }

 public static void method(List<Double> list) {
 System.out.println(list);
 }
}
```

- a.[22,23]
- b.[23,22]
- c.compilation error
- d.[22.0,23.0]

**Answer: Compilation error**

17.assume we have declared a static method with same name in the two interfaces and a class tries to implement both the interface. what is the result?

- a.the code will not be compiled due to static method of an interface cannot be overridden
- b.the code will not be compiled as two interface has same name for the static method
- c.the code will not be compiled we have to override the static method in the class to avoid the diamond problem
- d.the code will compile successfully

**Answer: the code will not be compiled due to static method of an interface cannot be overridden**

18.which of the following is incorrect way of declaring the variable 'employeeData'?

- a.Array<Employee> employeeData[] = new String[5];
- b.List<Employee> employeeData = new ArrayList();
- c.Map<Integer, String> employeeData = new LinkedHashMap<>();
- d. Set<Employee> employeeData = new HashSet<>();

**Answer: Array<Employee> employeeData[] = new String[5];**

19. //assume all the imports are added

```
Public class TestDemo {
 @Test
 Public void test() {
 Fail("Not yet implemented"); //line 1
 assertSame(1,2); //line 2
 }
}
```

What will be output when the above test case executed?

- a. test case passes
- b. test fails as fail() is meant for failing the test case explicitly
- c. test fails because of AssertionError in line 2
- d. compilation error at line 1

**Answer: compilation error at line 1**

20. what is the scope/access specifier of getEmployeeId() method in the code snippet given below?

```
Class Employee {
 Int employeeId;
 double getEmployeeId() {
 System.out.println("Employee Id");
 Return employeeId;
 }
}

a. public
b. private
c. protected
d. default
```

**Answer: private**

21. identify the output:

```
public class MyDemo {
 public static void main(String args[]) {
 int i = 5;
 switch(i) {
 case 1:
 System.out.println("One");
 Break;
 case 2://line1
 case 3://line 2
 System.out.println("Two and Three");
 case 4,5://line 3
 System.out.println("four and five");
 }
 }
}
```

```

 Break;

default
 System.out.println("Default");
}
}
}

a. compilation error in line 1 as there is no body for this case
b. compilation error in line 2 as there is no break statement
c. compilation error in line 3 as multiple values are not allowed in case
d. it will work fine and display "four and five" as output

```

**Answer: It will work fine and display "four and five" as output**

22. what is the result of attempting to compile and run this program?

```

class Demo {
 public static void main(String[] args) {
 float f1= new float(67,65f);//line 3
 float f2= new float(36,45f);//line 4
 if(f1>f2) {
 System.out.println("f1 is bigger than f2");
 }
 else {
 System.out.println(f1 is not bigger than f2");
 }
 }
}

```

- a. f1 is bigger than f2
- b. f1 is not bigger than f2
- c. no output

d. null

**Answer: f1 is bigger than f2**

23. Consider the below code snippet:

```
String[] customers=new String[]{"John", "Jackline", "Jasmine"};
```

```
List customerList= Array.asList(customers);
```

Select the best code to be used to iterate through the “customerList” object data.

1. 

```
for(iterator itr=customerList.iterator();itr.hasNext(); {
 System.out.println(itr.hasNext());
}
```
  2. 

```
for(String s:customerList)
 System.out.println(s);
```
  3. 

```
for(iterator itr=customerList.iterator();itr.next();)
 System.out.println(s.next());
}
```
  4. 

```
for(iterator itr=customerList.iterator(customerList.size());itr.hasPrevios();)
 System.out.println(s.previos());
}
```
- a. option 1  
b. option 2  
c. option 3  
d. option 4

**Answer: Option 2**

24. Given:

```
class Invoice {
 int count = 100;
 void billNo() throws Exception {
 try {
 count++;
```

```

 throw new Exception();
 } catch(Exception ex) {
 Count++;
 }
}

Void display() {
 System.out.println("Bill no." +count);
}

public static void main(String[] args) throws Exception {
 invoice inv = new invoice();
 inv.nillNo();
 inv.display();
}
}

```

Predict the output?

- a. Bill no: 103
- b. Bill no: 102
- c. Bill no: 101
- d. Bill no: 100

**Answer: Bill no: 102**

25. which of the below 'if' statement can be used to find a year is a leap year or not?

- a. if(((year % 4==4) && (year%100!=0)) || (year%400==4))
- b. if(((year % 4==0) && (year%100!=0)) || (year%400==0))
- c. if(((year % 4==0) && (year%100!=4)) || (year%400==4))
- d. if(((year % 4==4) && (year%100!=0)) || (year%400==0))

**Answer: if(((year % 4==0) && (year%100!=0)) || (year%400==0))**

26. which of the below is NOT a good practice for creating objects?

- a. lazy initialization of objects
- b. creating String literals instead of String objects
- c. creating Wrapper objects instead of primitives
- d. invoking static factory methods for immutable classes

**Answer: invoking static factory methods for immutable classes**

27. which of the below code is implemented without best practices standard?

```
1. String[] str=new String[]{"Hi","Hello","Welcome"};
List strList=Arrays.asList(str);
for(iterator itr=strList.iterator();itr.hasNext();) {
}
```

2. Integer i1=new Integer(11);

```
Integer i2=new Integer(11);
System.out.println(i1==i2);
```

- a. option(1) doesn't follow best practices can be improved using for(String s:strList)
- b. option(2) doesn't follow best practices as objects should not be compared with ==
- c. option(1) and (2) are not following best practices
- d. option (1) and (2) are following best practices

**Answer: option(1) and (2) are not following best practices**

28. public class OperatorDemo {

```
public static void main(String[] args)
{
 int x= 120, y= 110;

 String s1 = "Thank",s2="Thank";

 int arr1[] = {1,2,3};
```

```

int arr2[] = {1,2,3};
boolean boo = true;
System.out.println("x==y"+(x==y));
System.out.println("x<=y"+(x<=y));
System.out.println("s1==s2"+(arr1==arr2));
System.out.println("boo=true"+(boo==true));

}
}

a. x==y.false
 x<=y.false
 s1==s2.true
 boo==true false

b. x==y.false
 x<=y.false
 s1==s2.true
 boo==true true

c. x==y.true
 x<=y.false
 s1==s2.false
 boo==true true

d. x==yfalse
 x<=yfalse
 s1==s2false
 boo=true true

```

**ANSWER:**

**x==yfalse**  
**x<=yfalse**  
**s1==s2false**  
**boo=true true**

29.Which among the following option are correct with respect to HashMap(Select 2)

- a) Override Boolean equals(Object o)
- b) Override toString()
- c) Override int hashCode()
- d) Override String hashCode()

**Answer:Override Boolean equals(Object o), Override int hashCode()**

30.

```
public class Movie {
 private String movieName;
 public Exam(String name) {
 this.movieName=name;
 }
 @Override
 public boolean equals(Object obj) {
 return true;
 }
 Public class CollectionTester {
 public static void main (String [] args) {
 HashMap<Exam,String> hashMap = new HashMap<>();
 hashMap.put(new Exam("Harry Potter1"),"movie1");
 hashMap.put(new Exam("Harry Potter"), "movie2");
 hashMap.put(new Exam("Harry Potter1"), "movie3");
 System.out.println(hashMap.size());
 System.out.println(hashMap.get(new Exam("Harry Potter")));
 }
 }
}
```

- a. 1
- null
- b. 3
- null
- c. 2
- movie3

**ANSWER: 3 Null**

31.What will happen if the following code is executed?

```
Import.java.util;
public class TestDemo {
 public static void main(String[] args) {
 List<String> list1 = new ArrayList<>();
```

```

list1.add("1");
list1.add("2");
list1.add(1,"3");//Line 1
List<String> list2 = new LinkedList<>(list1); // Line2
list1.addAll(list2);//line3
list2 = list1.subList(2,6);//line4
list2.clear();
System.out.println(list1+"");
}
}
a. Run time error at line4 due to IndexOutOfBoundsException
b. {1,2,3}
c. {1,3,2}
d. {1,3{1,3,2}}

```

**ANSWER:** {1,3{1,3,2}}

**32.**

```

public static void main (String [] args) {
 TreeSet set = new TreeSet<>();
 set.add("a");
 set.add(6);
 set.add("c");
 Iterator ite = (Iterator) set.iterator();
 while(ite.hasNext()) {
 System.out.println(ite.next()+" ");
 }
}

```

Predict the output?

- a. a 6 c
- b. a followed by exception
- c. The code will give compile time error cannot add String
- d. The code will give java.lang.ClassCastException incompatible with java

**ANSWER:** The code will give Compile time error cannot add string

**33.** Predict the output?

```

//assume all the required imports are added
public class TestDemo {

```

```
@Test
public void test() {
 String s=" ";
 Assert.assertNotNull(a);
}
}

a. Test passes
b. Test fails
c. An exception is thrown at runtime
d. Compilation fails
```

**Answer: Compilation fails**

34.//assume all the required imports are added

```
public class TestDemo {
 private ArrayList list;//line1
 @Test//line2
 Public void test() {
 assertTrue(list.isEmpty());
 }
}
```

Which of the following statement is true regarding the above code fragment?

- a. If we replace line2 by '[@test\(expected=ArrayIndexOutOfBoundsException.class\)](#)' the test case will pass
- b. If we replace line2 by '[@test\(expected=NullPointerException.class\)](#)' the test case will pass
- c. If we replace line1 by 'privateArrayList list=new ArrayList<>();' the test case will pass
- d. test case will pass because the list is empty

**Answer: If we replace line1 by 'privateArrayList list=new ArrayList<>();' the test case will pass**

35.Which of the following doesn't support "this" keyword?

- a. Default and parameterized constructors

- b. Static blocks and methods
- c. Non static and private methods
- d. Abstract class

**Answer: Static blocks and methods**

36.Which of the below methods can be defined in a single class in java? Select the most suitable option

- 1. Void add(int, int)
  - 2. Void add(int, int, int)
  - 3. int add(int, int)
- 
- a. 1 and 3 together
  - b. 1 and 2 together
  - c. 2 and 3 together
  - d. 1,2 and 3 together

**Answer: 1 and 2 together**

37.Consider the below code snippet:

```
Calculator.java
Public class Calculator {
protected String add(int a, int b) {
return a+b;
}
}
```

Identify the option to write the TestCase for the add method?

- a. Test case cannot be written because add() is protected
- b. Test case should be in the different package of Target Class as public void testAdd()
- c. Test case should be in the same package of Target Class as public void testAdd()
- d. Test case should be in the same package of Target Class as protected void testAdd()

**Answer: Test case should be in the same package of Target Class as protected void testAdd()**

38.Given:

```
public class OperatorDemo {
 public static void main(String[] args) {
 int i1=15;
 String b1=i1>20?"Green":i1>10?"Blue":"Violet";
 System.out.println(b1);
 }
}
```

What will be the output?

- a. No output
- b. Green
- c. Blue
- d. Violet

**Answer:Blue**

39.Which of the following is false regarding Errors in java Exception handling?(any 2)

- a. An error shows a serious problem which should not be caught by an application
- b. An error usually occurs due to the lack in any of the system resources
- c. Errors can be handled by the code of the program
- d. An error is taken as the unchecked exception

**Answer: An error usually occurs due to the lack in any of the system resources**

**Errors can be handled by the code of the program**

40. Select the outcome of the below code snippet:

```
public class Demo {
 public static void main(String[] args) {
 int i=34;
 int j=34;
 System.out.println(i==j);
 Integer i1=new Integer(34);
 Integer i2=new integer(34);
 System.out.println(i1==i2);
 }
}
```

- a. True true
- b. True false
- c. True runtime error[wrapper classes cant be compared using==] will be thrown
- d. Compilation error [wrapper classes cant be compared using==] will be thrown

**Answer: Compilation error [wrapper classes cant be compared using==] will be thrown**

41. Which of the following statement is correct about singleton design pattern?

- 1. This type of design pattern comes under creational pattern
  - 2. This pattern involves a single class which is responsible to create an object which make sure that only single object gets created
  - 3. Singleton class provides a way to access the object directly without the need to instantiate the object of the class
- 
- a. Only statement 1
  - b. Only statement 2
  - c. Only statement 3
  - d. All the statements are true

**Answer:Only statement 2**

42. which of the following can be written in line 1 as per best coding practices?

```
public class Test {
 String[] arr={"infy","java", "ETA"};
 public void show() {
 for(int i=0;i<arr.length;i++)
 //line 1
 {
 System.out.println("name of a language");
 } else {
 System.out.println("language not found");
 }
 }
 public static void main(String[] args) {
 Test t=new Test();
 t.show();
 }
}
```

- a. If(arr[i].length()==4 & arr[i].equals("java"))
- b. If(arr[i].length()==4 | arr[i].equals("java"))
- c. If(arr[i].length()==4 && arr[i].equals("java"))
- d. If(arr[i].length()==4 || arr[i].equals("java"))

**Answer:If(arr[i].length()==4 && arr[i].equals("java"))**

43. Given

```
Interface Greeting {
 default void greet() {
```

```
System.out.println("In Greet interface");
}

}

class GreetingsDef implements Greeting {
public void greet() {
System.out.println("In GreetingDef class");
}
}

} public class App {
public static void main(String par[]) {
Greeting obj=new GreetingDef();
Obj.greet();
}
}
```

a. In Greet interface

In GreetingDef class

b. In GreetingDef class

c. In GreetingDef class

In Greet interface

d. In greet interface

**Answer: In GreetingDef class**

**In Greet interface**

44. Given

Util.java

```
public class Util {
```

```
public String concatenate(String str1, String str2) {
 return str1+str2;
}
}
}
```

UtilTest.java

```
Public class UtilTest {
 @Test
 //line1 {
 //set of statements
}
}
}
```

Identify the options to be used in line1 that gives java.lang exceptions?(any 2)

- a. void testConcatenate()
- b. public void testConcatenate()
- c. static void testConcatenate()
- d. public String testConcatenate()

**answer: static void testConcatenate(), public String testConcatenate()**

45. Given

```
Public static void main(String[] args) {
 Int twoD[][]=new int[4][];//line 1
 twoD[0]=new int[1];
 twoD[1]=new int[2];
 twoD[2]=new int[3];
 twoD[3]=new int[4];
 for(int i=0;i<4;i++) {
 for(int j=0;j<i+!;j++) {
 twoD[i][j];//line2
```

```
}

System.out.println("executed");

}

}

}
```

- a. compilation error in line 2 as there is no assignment operator used
- b. compilation error in line 1 as second index of array is not provided
- c. code gets executed and displays output as executed
- d. code gets executed but displays no output.

**Answer:compilation error in line 2 as there is no assignment operator used**

46. how many test cases will pass?

```
public class AssertChecker {

String first="abc";

String second="abc";

String third=new String("abc");
String fourth=new String("abc");
@Test
public void StringAssertEqual() {
Assert.assertEquals(first, second);
}

@Test
public void StringAssertEquals() {
Assert.assertEquals(third, fourth);
}

@Test
public void StringAssertSame() {
Assert.assertSame(first, second);
}

@Test
public void StringAssertSames() {
Assert.assertEquals(third, fourth);
}
```

}

- a. all 4 will pass
- b. only first 3 will pass
- c. only first 2 will pass
- d. none of the testcases will pass

**answer: all 4 will pass**

47. what changes need to be made in the following code to make the singleton pattern correct? (any 2)

```
public class Employee {
 public static Employee employeeInstance;
 private Employee() {
 public static Employee getEmployee()
 {
 If(employeeInstance==null) {
 employeeInstance=new Employee();
 }
 Return employeeInstance();
 }
 }
}
```

- a. none, the singleton pattern is properly implemented
- b. rename employee to instance
- c. add synchronized to getEmployee()
- d. change the access modifier of employeeInstance from public to private

**answer: add synchronized to getEmployee()**

48. From the below options, identify the methods and constructors in throwable that support chained exceptions:

1. throwable getCause()
2. throwable initCause(throwable)
3. throwable(String, throwable)
4. throwable(throwable)

- a. option 3 and 4 only
- b. option 1,2,3,4
- c. option 4 only
- d. option 1 and 3, and 4 only

**answer: option 3 and 4 only**

49. Consider the below code snippet:

```
Locale locale=new Locale("da","DK");
NumberFormat numberFormat=NumberFormat.getInstance(locale);
String number=numberFormat.format(100,99);
```

```
System.out.println(number);
```

Which among the following design pattern is followed by NumberFormat.getInstance() ?

- a. factory method pattern
- b. singleton pattern
- c. prototype pattern
- d. builder pattern

**answer: factory method pattern**

50. what is the output when the below code is compiled and executed?

```
public class Customer {
 public static void main(String[] args) {
 int numbers[] = new int[2];
 numbers[0] = 1265;
 numbers[1] = 1212;
 int j = 0;
 for (int i = 0; i < numbers.length; i++) {
 System.out.println(numbers[j]);
 ++j;
 }
 }
}
```

- a. 1265  
    1234
- b. 1265  
    1265
- c. 1265  
    1212

**Answer:1265**

**1212**

51. Consider the following code:

```
Interface Parking {
 Static void Prk() {
 System.out.println("from parking interface");
 }
}
public class CarParking implements Parking {
 static void park() {
```

```
 system.out.println("from CarParking class");
 }
public static void main(String[] args) {
 Parking.park();
 Park();
}
}
```

What will be the output of the code after execution?

- a. From Parking Interface  
From CarParking class
- b. From Parking Interface  
From Parking Interface
- c. Compilation error at line 1 as park() method should be called in static way
- d. compilation error at line 1 as park() method should be called with a reference variable

**answer: From Parking Interface  
From CarParking class**

52. public class LoopDemo {  
 public char display(int marks) {  
 if(marks>=85 && marks<99) {  
 return'A';  
 else if(marks>65 && marks <= 84) {  
 return 'B'; }  
 else {  
 return'F';  
 }  
 }  
}

TestLoop.java  
public class TestLoop {  
 LoopDemo tc=new LoopDemo();  
 Char value;  
@Test  
public void testdisplay() {  
 Assert.assertSame(tc.display(70), 'B');  
//line1  
}  
}

To get 100% branch coverage which code should be added in line 1?

- a.Assert.assertSame(tc.display(88), 'A');  
Assert.assertSame(tc.display(64), 'F');

```
Assert.assertSame(tc.display(100), 'F');

b.Assert.assertSame(tc.display(88), 'A');
Assert.assertSame(tc.display(64), 'F');

c. Assert.assertSame(tc.display(85), 'A');
Assert.assertSame(tc.display(64), 'F');
Assert.assertSame(tc.display(84), 'B');

d.Assert.assertSame(tc.display(85), 'B');
Assert.assertSame(tc.display(84), 'F');
Assert.assertSame(tc.display(74), 'A');
```

**Answer:** **Assert.assertSame(tc.display(88), 'A');**  
**Assert.assertSame(tc.display(64), 'F');**

# JAVA SE8 DEVELOPER CERTIFICATION

1. Class AccessModifier

```
{
 public int y = 20;
 float z = 30;
 protected int a = 20;
 private int b = 20;
}
```

Identify the order of the variables based on the visibility from high to low.

- a. y a z b
- b. a z b y
- c. z a b y
- d. a b y z

2. class Parent{

```
 public final void show(){
 System.out.println("show() inside Employee");
 }
}
```

final class Child extends Parent{

```
 public void show1(){ //Line 1
 final int x = 100;
 System.out.println("show() inside Unit");
 System.out.println(x);
 }
}
```

public class Demo{

```
 public static void main(String[] args){
 Parent parent = new Child();
 new Child().show1(); //Line 2
 }
}
```

What will be the output when the above code is compiled and executed?

Answer : show() inside Unit

100

```
3. class Greeter{
 public static void main(String[] args) throws Exception {

 try{
 System.out.println("Greetings!!"+ " "+args[0]);
 }catch(ArrayIndexOutOfBoundsException e){
 System.out.println("Sam");
 }
 }
}
```

Predict the output?

- a. Greetings!! Sam
- b. Sam**
- c. Compilation error
- d. Runtime error

4. class Expression

```
{
public void calc(){
 double x = 10;
 int y = 20;
 float z = 30;
 //line 1
 b = x+y+z;
}}
```

Identify the suitable datatype to be used for variable “b” at line 1?

- a. Long
- b. Int
- c. Double**
- d. Float

5. Which of the below is NOT a good practice for creating objects?

- a. Lazy initialization of objects
- b. Creating String literals instead of String objects
- c. Creating wrapper objects instead of primitives**
- d. Invoking static factory methods for immutable classes

```

6. class Invoice{
 int count = 100;
 void billNo() throws Exception{
 try{
 count++;
 throw new Exception();
 }
 catch(Exception ex){
 count++;
 }
 }
 void display(){
 System.out.println("Bill no : "+count);
 }
 public static void main(String[] args) throws Exception{
 Invoice inv = new Invoice();
 inv.billNo();
 inv.display();
 }
}

```

Predict the output : Answer -> Bill no : 102

```

7. class Book{
 public static void main(String[] args) throws Exception{
 System.out.println("Invoke "+jsp());
 }
 static String jsp(){
 try{
 throw new IOException(); }
 catch(IOException e){
 return "IO"; }
 finally{
 return "Finish";
 }}}

```

Predict the output

- a. Invoke IO
- b. Invoke IO  
Invoke Finish
- c. Invoke Finish
- d. The code will give compilation error as jsp() has not returned any value

```
8. public class WrapperClassDemo{
 public static void main(String aa[])
 {
 Integer intWrapper = Integer.valueOf("12345"); //line 1
 Integer intWrapper2 = Integer.valueOf("11",2); //line 1
 Integer intWrapper3 = Integer.valueOf("E",16); //line 1
 System.out.println(intWrapper+ " "+intWrapper2+ " "+intWrapper3);
 }
}
```

Predict the output

- a. 12345 11 E
  - b. 12345 11 14
  - c. 12345 3 14
  - d. Runtime Error in Line 3 due to NumberFormatException
9. Which of the following pattern refers to creating duplicate object while keeping performances in mind?
- a. Builder Pattern
  - b. Bridge Pattern
  - c. Prototype Pattern
  - d. Filter Pattern
10. Which of the following is false regarding super keyword in Java?
- a. "super" keyword can be used to call a parent class public constructor which is present in a different package
  - b. "super" keyword can be used to call a parent class private constructor which is present in the same package
  - c. "super" keyword can be used to call a parent class protected constructor which is present in the same package
  - d. "super" keyword is used to call a parent class protected constructor which is present in a different package
11. Which among the following is valid option for wildcards?(Select 2 options)
- a. Used to relax restriction on the variable
  - b. Used in scenario where type being operated upon is not known
  - c. Used in generic method type argument
  - d. Can access members of super class

```
12. class Message{
 public static void main(String[] name) throws Exception {
 Integer[][] val = { {10,12,14}, {null}, {18,20,22} };
 System.out.println("Array of = "+val[1][1].intValue());
 }
}
```

Predict the exception that occurs?

- a. StringIndexOutOfBoundsException : change the val data type to String[]
- b. RuntimeException : due to inappropriate use of null
- c. ArrayIndexOutOfBoundsException : there is no integer values in the index[1][1]

13. Which of the following statements are correct w.r.t Java version 8 (select two)

- a. The JDK will be installed as part of the JRE installation
- b. The JRE includes the Java tool which launches the JVM to execute the Java program
- c. JDK can also be installed without JRE
- d. JRE installable's are available for different platforms like Windows and Linux

14. Which among the following comes under Creational Design Pattern?

- a. Iterator Design Pattern
- b. Singleton Design Pattern
- c. Command Pattern
- d. Observer Design Pattern

15. public class Sample{

```
 public static void main(String[] args){
 Integer x = new Integer(89);
 long y = x.longValue();
 System.out.println(y);
 }
}
```

Predict the output

- a. 89.0
- b. 89L
- c. 89
- d. 89.00000

16. Which of the following are the correct way to declare a generic method in Java?(choose any 3)

- a. access-specifier <generic-type-parameter-list> return-type method-name(parameter-list) { }
- b. access-specifier static-keyword <generic-type-parameter-list> return-type method-name(parameter-list) { }
- c. access-specifier return-type <generic-type-parameter-list> method-name(parameter-list) { }
- d. <generic-type-parameter-list> return-type method-name(parameter-list) { }

17. Consider the following statements

- 1. java.util.Calendar class implements factory design pattern
- 2. java.util.ResourceBundle.getBundle() implements factory design pattern
- 3. java.awt.Desktop.getDesktop() implements Singleton pattern design pattern

Which of the above mentioned statements are true?

- a. All the statements
- b. Only 1
- c. Only 2
- d. Only 3

18. What is the result of attempting to compile and run this program?

```
class CustomException extends Exception{
}
Class Customer extends CustomException{
}
public class ExceptionDemo {
 public static void main(String args[]) {
 try{
 throw new Customer();
 } catch(CustomException customException){
 System.out.println("CustomException catch block");
 } catch(Customer customer){
 System.out.println("Customer catch block");
 }}}
```

a – Prints “CustomerException catch block”

b – Prints “Customer catch block”

c – Compilaton error because customer class exception is not throwable

```
19. public class Customer{
 public static void main(String[] args) {
 int numbers[] = new int[2];
 numbers[0] = 1265;
 numbers[1] = 1212;
 int j=0;
 for(int i=0; i<numbers.length; i++){
 System.out.println(numbers[j]);
 ++j;
 }
 }
}
```

Predict the output

- a. 1265  
1234
- b. 1265  
1265
- c. 1265  
1212

```
20. public class Test{
 public static void main(String[] args) {
 int [][] x;
 x = new int[3][4];
 for(int i=0; i<3; i+=2) {
 for(int j=0; j<4;j++) {
 x[i][j] = i + j;
 System.out.println(x[i][j]+ " ");
 }
 }
 }
}
```

Predict the output

- a. 0 1 2 3 1 2 3 4 2 3 4 5
- b. 0 1 2 3 2 3 4 5
- c. 0 2 1 3 2 4
- d. 0 2 2 4

21. All the wrapper Classes (Character, Boolean, Byte, Integer, Double, Long, Short, Float) extends which interface/class ? (Select 2)

- a. Serializable
- b. Number
- c. Comparable
- d. BufferedReader

22. Which among the following code snippets are illustrating the concept of autoboxing here? (Choose 2)

- a. Character ch = 'a';
- b. ArrayList<Integer> listOfTickets = new ArrayList<>();  
listOfTickets.add(101);  
listOfTickets.add(102);
- c. Int var1 = new Integer(1003);
- d. ArrayList arrayList = new ArrayList();  
Int number = arrayList.get(0);

23. Which of the following is not part of Throwable hierarchy in Java?

- a. OutOfStackError
- b. OutOfMemoryError
- c. VirtualMachineError
- d. StackOverFlowError

24. From the below options identify the methods and constructors in Throwable that support chained exceptions.

- 1. Throwable getCause()
  - 2. Throwable initCause(Throwable)
  - 3. Throwable(String, Throwable)
  - 4. Throwable(Throwable)
- 
- a. Option 3 & 4 only
  - b. Option 1,2,3,4
  - c. Option 4 only
  - d. Options 1,3,4 only

25. Which of the following method of Assert class checks that a condition is false?

- a. void assert(boolean condition)
- b. void assertFalse(boolean condition)
- c. void assertCheck(boolean condition)
- d. void assertChecks(boolean condition)

26. Which of the following is the correct usage of a relational operator inside an if statement?

- a. if(firstName == "Annie")
- b. if(firstName.equals("Annie"))
- c. if(firstName.equals("Annie")) && salary==50000)
- d. if(firstName == "Annie" | !salary==50000)

27. How can you find out the difference between two dates?

- a. Date1.difference(date2)
- b. Date1.until(date2,TemporalUnit);
- c. Date1.between(date2);
- d. Date1.minus(date2);

28. public class Sample{

```
public static void main(String par[]){
 LocalDateTime date = LocalDateTime.of(2020,06,7,1,1,1).plusHours(30);
 DateTimeFormatter formatterPattern = DateTimeFormatter.ofPattern("dd-MMM-yyyy");
 String str = date.format(formatterPattern);
 System.out.println(str);
}
```

Predict the output?

- a. 08-06-2020
- b. 07-Jun-2020
- c. 08-Jun-2020
- d. 09-06-2020

29. public class Demo{

```
private static String id;
private Random random = new Random();
public Demo() {
 //line 5
 id = "ACC1101" + Math.abs(random.nextInt());
}
public String getId() {
 return id;
}
```

- a. if(id == null) {
- b. synchronized(this) if(id == null) {
- c. if(id == null){synchronized(this){
- d. synchronized{if(id==null){

Java:

1. What will be the output of the below code?

```
//Assume all the required import statements are added
Public class TestClass
@TesT
Public void test(){
String a = "";
AssertSame("a",a);
```

**Ans: An exception is thrown at runtime as assertSame() is not a valid function**

2. What will be the output of the below code?

```
Public class Demo{
 Public static void main(String args[]){
 Int i = 34;
 Int j = 34;
 System.out.println(i==j);
 Integer i1 = new Integer(34);
 Interger i2 = new Integer(34);
 System.out.println(i1==i2);
 }
}
```

**Ans: true false**

3. Which out of the following are true in regard to interfaces in Java?

- (i) An interface can contain default and static method with defined bodies.
- (ii) An object can be created of an interface
- (iii) Multiple inheritance is allowed in Interfaces
- (iv) Multi-level inheritance is not possible in interfaces

**Ans: (i),(iii) & (iv)**

4. Given

```
Public class Sample{
 Public static void main(String[] args) throws Exception{
 Try{
 System.out.println("In try block");
 System.exit(0);
 }catch(Exception ex){
 System.out.println("In catch block");
 Ex.printStackTrace();
 }finally{
 System.out.println("In finally block");
```

```
 }
 }
}
```

Predict the output?

Ans: In try block

5. Which of the following are the advantages of exception handling in Java(Choose any 3 options)?
- (i) To maintain the normal flow of execution
  - (ii) Meaningful error reporting
  - (iii) To document compile time error
  - (iv) To prevent the abrupt termination of a program

Ans: (i),(ii) &(iv)

6. Predict the output of the below code:

```
final class Square{
 private double length,breadth;
 public Square(double length,double breadth){
 this.length = length;
 this.breadth = breadth;
 }
 Square(Square square)
 {
 System.out.println("Copy constructor invoke");
 length = square.length;
 breadth = square.breadth;
 }
 public String toString(){
 return "("+length+"+"+breadth+"i)";
 }
}
class Main{
 public static void main(String[] args){
 Square square1 = new Square(110,115);
 Square square2 = new Square(square1);
 System.out.println(square2);
 }
}
```

Ans: Copy constructor invoked  
(110.0 + 115.0i)

7. Which of the following is a valid lambda expression?

Ans: (a,b) -> {int result; return result>0;}

8. What is the output of the following code?

```
Class Employee{
 Void display(char c){
 System.out.println("Employee name starts with: " +c+ ".");
 System.out.println("His experience is : 11 years.");
 }
}

Class Main extends Employee{
 Void display(char c){
 Super.display();
 System.out.println("Another employee name also starts with: " +c+ ".");
 New Employee().display('D');
 Display(7);
 }
 String display(int c){
 System.out.println("His experience is: "+c+" years.");
 Return "Bye";
 }
}

Public class Demo{
 Public static void main(String a[]){
 Employee emp = new Main();
 Emp.display('S');
 }
}

Ans: Employee name starts with : S.
 His experience is : 11 years.
 Another employee name also starts with : S. Employee name starts with : D.
 His Experience is : 11 years.
 His Experience is : 7 years.
```

9. What is the output of the below code?

```
Import java.util.HashSet;
Public class TestDemo{
 Public static void main(String...sss){
 HashSet myMap = new HashSet();
 String s1 = new String("das");
 String s2 = new String("das");
 NameBean s3 = new NameBean("abcdef");
 NameBean s4 = new NameBean(:abcdef");
 myMap.add(s1);
 myMap.add(s2);
 myMap.add(s3);
 myMap.add(s4);
 System.out.println(myMap);
 }
}
Class NameBean{
 Private String str;
 NameBean(String str){
 This.str = str;
 }
 Public String toString(){
 Return str;
 }
}
Ans: [abcdef,das,abcdef]
```

10. Which among the following is/are true about Design Pattern

- i. Design pattern depends upon abstraction
- ii. Design patterns are completed designs that can be transformed directly into code
- iii. Design pattern depends on abstraction, follows the process of Dependency Injection
- iv. Design pattern is a template of solving problem that can be used in many real world software development

Ans i & ii

11. From the below options identify the methods and constructors in Throwable that support chained exceptions

- i. Throwable.getCause()
- ii. Throwable.initCause(Throwable)
- iii. Throwable(String, Throwable)
- iv. Throwable(Throwable)

Ans : Options i, iii & iv

12. What is Magic Number in Java in the context of java programming best practice

- i. A number which gets printed on the console
- ii. A direct usage of the number in the code
- iii. A number which magically disappears from the code
- iv. A number which is generated through error

Ans: Option2

A direct usage of the number in the code

13. Default format of LocalDate is

- I. yyyy,mm,dd
- II. yyyy.mm.dd
- III. yyyy.dd.mm
- IV. yyyy,dd,mm

Ans Option ii

14. Which among the following option are correct with respect to HashMap

- I. Override boolean equals(Object o)
- II. Override toString()
- III. Override int hashCode()
- IV. Override String hashCode()

Ans i, Override boolean equals(Object o)

15. Which of the foll interfaces are not a part of java collections framework(choose any2)

- i. List
- ii. Queue
- iii. SoretdList
- iv. ArrayList

Ans: Option iii & iv

16. Where are string objects stored in memory using “new” keyword?

- i. In Stack Memory
- ii. In String constant pool in Heap Memory
- iii. In Native Methods stack Memory
- iv. Anywhere in Heap Memory

Ans iv

17. What will be written at the line1 so that the below code will compile and run successfully?

```
Public class Main{
 //line1
 static {
 X[0] = 102;
 }
 public static void main(String [] args){
 System.out.println(x[0]);
 }
}
```

Ans : option ii, static int[]x = new int[3];

18. Given

```
public class Employee{
 private String roles;
 Public Employee (String role) {this.role= role;}
 Public Boolean equals (Employee emp){return emp.role.equals(this.role)}
 }
}
```

Which among the following are correct with respect to the above code(Choose 2)

- I. Code give compilation error due to private attribute emp.role cannot be accessed
- II. Object.equals() method is not properly overriden
- III. Code will compile successfully
- IV. hashCode() method implementation is required

Ans iii & iv

19. Output of the below code

```
Public class TestDemo{
 Public static void main(String [] args){
 Integer n1 = new Integer (100);
 Integer n2= new Integer (100);
 Integer n3 =127;
 Integer n4 =127;
 System.out.println(n1==n2)
 System.out.println(n3==n4)
 }
}
```

Ans: false, true (option1)

20. Given

```
Public class ExceptionDemo{
 Static class Car implements AutoCloseable{
 Public void close(){
 System.out.println("Car door close")
 Throw new RuntimeException();
 }
 }

 Public static void main (String [] args){
 Try{
 //Line1

 }catch (Exception e){

 System.out.println("catch exception");
 }finally{

 System.out.println("finally")
 }
 }

 I. Car car = new Car();
 System.out.println("try block")

 II. Car car = new Car();
 Car.close();
 System.out.println("try block")

 III. Car car = new Car();

 IV. System.out.println("try block")
```

Ans: all four

21. Given an abstract class Customer as below

```
Public abstract class Customer{
 Public abstract String getCustomerType();
}
```

Select a valid implementation of getCustomerType method in another class from the below options

a. Abstract class C1 extends Customer{  
 Public abstract String getCustomerType(){ return "Premium";}  
}

- b. Customer customer = new Customer(){
 Public String getCustomerType(){ return "Premium";}
 }
  
- c. class C1 extends Customer{
 Public abstract String getCustomerType(){ return "Premium";}
 }
  
- d. new Customer(){
 return "Premium";
 }

**Ans: a & c**

22. Given below

```

Public class Demo{
@BeforeClass
Public static void afterClass(){
System. Out.println(1)
}

@After
Public void before(){
System. Out.println(3)
}

@Test
Public void test(){
System. Out.println(5)
}

@Before
Public void fter(){
System. Out.println(4)
}
@AfterClass
Public static void afterClass(){
System. Out.println(2)
}

```

**Ans Option 1 :**

1  
4  
5  
3  
2

23. Which of the following is 'FALSE' regarding 'super' keyword in Java?

Option3, Super keyword can be used to call a parent class protected constructor which is present in the same package

24. Which of the foll data structure is used by varargs in java

Ans: Option3, ArrayList

25. Which of the following is 'FALSE' regarding 'this' keyword in Java?

Ans option4, 'this' keyword can be used to create a block of code.

26. How many number of values can be accommodated by the varargs in java

Option4, Any number of values

27. Which of the foll statement is FALSE(Choose 2 options)

- I. An interface can extend from only one interfaces
- II. A class can extend from another class and at the same time implement any number of interfaces
- III. A class can extend multiple abstract classes
- IV. Many classes can implement the same interface

Ans i and iii

28. Have a look at the foll code and choose the correct option

```
Class ReportUtil{
 Int reportId =0;
 Static {++reportId}
 Static int employeeReportId;
 Public void preReport(){
 employeeReportId= reportId;
 }
}
```

Ans: option 3 The code will not get compiled as instance variable cannot be referred from the static block

29. Which of the below methods can be defined in the single class in java. Select most suitable(select 2 options)

- 1. Void add(int,int)
- 2. Void add(int,int, int)
- 3. int add(int,int)
  - I. 1 & 3 together
  - II. 1 & 2 together
  - III. 2 &3 together
  - IV. 1,2 &3 together

**Ans Options ii and iii**

30. Ria has a class called Account.java under two separate packages com.infy.debit and com.infy.credit. Can she use the Account class of both the packages in another class called ReportUtil.java of package com.infy.util

**Option2: No, She cannot as there will be a compilation error stating that the import collides with another import**

31. Which of the following are true about enums in Java? (Choose 3 correct)

- I. enums can implement any other interface in java
- II. an instance of enum can be created outside of enum itself
- III. enum cant extend any other class except the abstract base class java.lang.enum
- IV. enum can be used to implement singleton design pattern

**Ans i, iii & iv**

32. What's is true with respect to abstract class being given below

```
Abstract class Employee{
 //fields and constructor
 Public void salaryComputer()
 {
 // code goes here
 }
 Public abstract void taxReduce()
 {
 // code goes here
 }
 Public abstract void benifitsInclude();
}
```

**Ans: option4. Abstract method of class Employee has definition**

33. Which of the following are NOT good practice for creating objects.

- a. Lazy initialization of objects
- b. Creating String literals instead of String objects
- c. **Creating Wrapper objects instead of primitives**
- d. invoking static factory methods for immutable classes

**Ans option c**

34. Which among the foll are valid lamda expressions to sort the numbers in numberlist in descending order choose 3

- a. numberList.sort((x,y)->x.compareTo(y))
- b. numberList.sort(int x,int y)->x.compareTo(y))
- c. numberList.sort((x,y)->{return x.compareTo(y)})
- d. numberList.sort((Integer x, Integer y)->x.compareTo(y))

Ans

35. What is the output for the below code

```
public class VarArgsDemo{
 static void func(int...x)
 {
 System.out.println("Number of arguments "+x.length);

 for(int i:x)
 System.out.print(i+"");
 System.out.println();
 }
 void func(int a) //Line1
 { System.out.println("one"); }

 public static void main(String[]args){
 new VarArgsDemo().func(150);
 func(11, 12, 13, 14);
 func(); }
 }
Ans:Option B
```

```
Number of arguments1
150
Number of arguments4
11121314
Number of arguments0
```

36.Given, Assume all the required imports are added

```
Public class TestDemo{
 Private ArrayList list //Line1
 @Test //lin2
 Public void test(){
 assertTrue(list.isEmpty());
 }
}
```

Which of the following is true regarding the above code fragment

```
Ans:if we replace Line2 by “@Test(expected = NullPointerException.class)” the test
case will pass
```

37. Given

```
HashMap <Integer, Integer> myMap = new HashMap<Integer, Integer>();
myMap.put(1001,5);
myMap.put(1002,8);
myMap.put(1002,5);
myMap.replace(1002,5, 100);
system.out.println(myMap)
```

```
Ans: {1001=5, 1002=100}
```

38. What is the result when the foll code is compiled?

```
Class Student extends Exception{}
Class Hosteller extends Student{}
Public class StudentTester{
 Public static void main (String args[]){
 Try{
 //some monitored code
 Throw new Hosteller();

 }Catch(Student st){
 System .out.println("Student class Exception");
 }catch (Hosteller host){
 System .out.println("Hosteller class Exception");
 }
 }
}
```

**Ans: The code will not compile Unreachable catch for Hosteller because Student class exception is caught before Hosteller**

39.

```
class Operations{
 Public void addition(){
}
```

```
Class AdvOperations extends Operations{
Void addition()//line1
```

```
{
}
```

Line 1 generates compilation error . which of then below option helps to resolve this

**Ans: public keyword has to be added**

40. What will be in line1 to get the output as 150. Choose 2

```
class EmployeeUtil {
 //|line1
 double amountTax;
 public double taxCalculator(double salary)
 {
 amountTax = salary *TAX_PERCENTAGE/100;
 return amountTax;
 }
}
public class ConstantMain
{
```

```

public static void main(String[] args) {
 System.out.println(new EmployeeUtil().taxCalculator(5000));
}
}

```

Ans:

```

final int TAX_PERCENTAGE=3;
static int TAX_PERCENTAGE=3;

```

41.

```

interface Component {
 String cname = "Motor";
 String getName(String name);
}

public class Demo implements Component{
 public String getName(String name) {
 System.out.println("Inside Demo Class");
 return "Componenet from interface is :" + cname+"and component
from class is "+ name+ ";";
 }

 public static void main(String[] args) {

 Demo demo = new Demo();
 System.out.println(demo.getName("Battery"));
 demo.getName("Battery");

 }
}

```

Ans Inside Demo Class  
 Componenet from interface is :Motorand component from class is Battery;  
 Inside Demo Class

42.

```

public class CollectionTest {

 public static void main(String[] args) {

 Collection<Integer> employeeCollection = new HashSet();
 collection.add(1001);
 collection.add(1002);
 collection.add(1001);

 Set<Integer> newSet = new TreeSet();
 }
}

```

```

 newSet.addAll(employeeCollection);
 System.out.println(newSet);

 }
}


```

**Ans : Runtime Exception : null pointer Exception**

43. Which of the following regarding an abstract class is/are true in Java

- i) Object of an abstract class cant be created
- ii) An abstract class is designedonly to act as a base class in hierarchy to be inherited by other classes

**Ans : Both i & ii**

44.

```

public class CodeForException {
 //public static void main(String[] args) {
 public void callMe() throws Exception{
 try
 {
 int value=3/0;
 }
 catch (ArithmeticeException aa)
 {

 System.out.println(aa);
 }
 }
 public void calling()
 {

 callMe();
 }
}

```

**Ans : Compilation error**

45.

```

public class Demo {
 private static String id;
 private Random random = new Random();

 public Demo(){

 if(id ==null){
 }
 }
}

```

```

 id = "ACC1101"+ Math.abs(random.nextInt());
 }
}

public String getid() {
 return id;
}

}
Ans - if(id==null){

```

**46. Given**

```

//Assume all the required imports are added

public class TestDemo {
 private ArrayList list;//Line1
 @Test //Line2 (expected = NullPointerException.class)
 public void test(){
 assertTrue(list.isEmpty());
 }
}

```

Which of the following statement is true regarding the above code fragment?

Ans: if we replace Line2 by “@Test(expected = NullPointerException.class)” the test case will pass

**47.**

```

public static void main(String args[])
{
 HashMap<Integer, Integer> myMap = new HashMap<Integer, Integer>();
 myMap.put(1001, 5);
 myMap.put(1002, 8);
 myMap.put(1002, 5);
 myMap.replace(1002, 5, 100);
 System.out.println(myMap);
}

```

Ans: {1001=5, 1002=100}

48. what is the result when the following code snippet is compiled?

```

class Student extends Exception{}
class Hosteller extends Student{}
public class StudentTester {
 public static void main(String[] args){
 try{
 //some monitored code
 throw new Hosteller();
 }
 catch (Student st){

```

```

 System.out.println("Student class exception");
 }
 catch (Hosteller host){
 System.out.println("Hosteller class exception");
 }
}
}

```

**Ans:** The code will not compile Unreachable catch for Hosteller because Student class exception is caught before Hosteller

49.

```

class Operations
{
 public void addition()
 {}
}
class AdvOperations extends Operations {
 void addition()//Line1
 {}
}

```

Line 1 generates Compilation error. Which of the below options helps to resolve this?

**Ans:** public keyword has to be added

50.

```

Public class TestDemo{

Public void main(int x){

System.out.println("Main1")

}

Public static void main(String args[]){

System.out.println("Hello Main")

}

}

```

**Ans :** option 4 : Hello Main

51.

```
import java.util.HashMap;
import java.util.Iterator;
import java.util.TreeSet;
public class ABC {
 public static void main(String args[])
 {
 TreeSet set = new TreeSet();
 set.add("a");
 set.add(6);
 set.add("c");
 Iterator ite =(Iterator) set.iterator();
 while(ite.hasNext()){
 System.out.println(ite.next()+"");
 }
 }
}
```

Ans: The code will give `java.lang.ClassCastException: java.lang.String incompatible with java.lang.Integer`

52. what is the output of the following code?

```
public class Demo5 {
 public static void main(String[] args){

 int x[][] = new int [4][]; //Line 1
 x[0] = new int [1];
 x[1] = new int [2];
 x[2] = new int [3];
 x[3] = new int [4];
 int a,b,c=0;
 for(a=0; a<4; a++){
 for(b=0; b<a+1; b++){
 x[a][b] = c;
 c++;
 }
 }
 for(a=0; a<4; a++){
 for(b=0; b<a+1; b++){
 System.out.print(" " + x[a][b]);
 c++;
 }
 }
 System.out.println();
 }
}
```

```
}
```

Ans:

```
0
12
345
6789
```

53. which of the following keyword is used to prevent the content of a variable from being modified from outside.

Ans - Final

54.

```
class Book{
 int bookid =2356;

}
class Book1 extends Book{
 int bookid = 1167;
}
class Book2 extends Book1{
 int bookid = 2378;//Line8
 void display(){
 System.out.println(super.super.bookid);//Line10
 System.out.println(super.bookid);//Line11
 System.out.println(bookid);
 }
}
public class Demo2 {
 public static void main(String[] args){
 Book2 book2 = new Book2();
 book2.display();
 }
}
```

Ans - Compilation fails, because of an error in Line10 as super keyword is unexpected.

55. Consider the below class and identify the extension of the output file when we execute the command javac Employee.java

```
class Employee1 {
 private int x=10;
 public void showX(){
 System.out.println(x);
 }
}
```

Ans - .java

56. which of the below if statement is used to find a year is a leap year or not

```
Ans: if((y%4==0) && (y%100 !=0) || (y%400==0))
```

57. Analyze the below code and predict the output when executed the code

```
public class Demo6 extends Book{
 int bookid = 4362;
 public int getvarchar(){
 return bookid;
 }
 public void Cell(){
 System.out.println(getvarchar()); //Line1
 }
 public static void main(String[] args){
 Book book = new Book();
 super.Cell(); //Line 2
 }
}
class Book{
 int bookid = 17252;
 public int getvarchar(){
 return bookid;
 }
}
```

```
Ans: Compilation error in line 2 as super keyword cannot be used in static context
```

58. have a look at the following class and predict what should be placed at line 1 to get 150.0 as output when the code gets executed? (choose 2 options)

```
class EmployeeUtil{
 //line1
 double amountTax;
 public double taxCalulate(double salary){
 amountTax = salary*TAX_PERCENTAGE/100;
 return amountTax;
 }
}
public class ConstantMain{
 public static void main(String[] args){
 System.out.println(new EmployeeUtil().taxCalulate(5000));
 }
}
```

Ans:

```
static int TAX_PERCENTAGE=3;
final int TAX_PERCENTAGE=3;
```

**59.** if the child class of an abstract class does not implement all its abstract methods then it should be declared as

**Ans: Option A: Abstract class**

60. Which of the following statements regarding an abstract class is/are true in Java

- I. Object of an abstract class cant be created
- II. An abstract class is designed only to act as a base class in hierarchy to be inherited by other classes

**Ans : Both I & II**

61. which of the following is valid function used to read values using Scanner in java (choose 3)

1. **nextInt()**
2. **nextChar()**
3. **nextLong()**
4. **nextLine()**

**Ans: option 1,3 and 4**

62. What will be output of the below code snippet?

```
package Example;

public class Pet {
 public void displayName(){
 System.out.println("Inside Pet");
 }
}

package java.pac;
import Example.Pet;
public class Dog extends Pet{
 @Override
 public void displayName(){
 System.out.println("Inside Dog");
 }
}
```

```

 }
}

package com.infy;
import Example.Pet;
public class Demo7 {
 public static void main(String[] args){
 Pet pet = new Dog(); //Line1
 pet.displayName();
 }
}

```

**Ans – Compilation error at Line1 due to ClassCastException**

63. Which of the foll is FALSE regarding polymorphism in JAVA

- a. Polymorphism is the ability of an object to take different forms
- b. Polymorphism can be as a single action that can be performed in different ways
- c. Polymorphism means different forms by creating different classes that are referred to each other by inheritance
- d. Polymorphism is using same method multiple times in a class with different parameters

**Ans Option D**

64. what is the result of the following code?

```

public class Vehicle {
 static class Car{
 public void go(){
 System.out.println("Car Ignition");
 }
 }
 static class ElectricCar extends Car{
 public void go(){
 System.out.println("ElectricCar Ignition");
 }
 }
 static class PetrolCar extends Car{
 public void go(){
 System.out.println("PetrolCar Ignition");
 }
 }
 public static void main(String[] args){
 Car car = new ElectricCar();
 car.go();
 }
}

```

**Ans- ElectricCar Ignition**

65. If child class of abstract class doesnot implement all its abstract methods then it should be declared as?

**Ans - abstract class**

66. which are valid upper bound by class Employee of list. (Doubt)

**Ans - List<?extends Employee>**

**Choose any 2**

```
public class App {
 public static void main(String[] args) {
 String msg = null;
 try {
 System.out.println(msg.length());

 }catch(NullPointerException ex) {
 System.out.println("Exception is caught here");

 throw ex; //Line 1
 System.out.println(msg); //Line 2
 }
 }
}
```

**Ans:**

**//a, b**

**A.Place line2 before line 1**

**B.Remove line 2 ie after throw, there should not be any statements**

```
interface Book {
 static void bookName() {
 System.out.println("in interface book");
 }
}

public class BookImpl implements Book{
 void bookName() {
 System.out.println("In BookImpl Class");
 }
}
```

```
}
```

```
public class BookApp {

 public static void main(String[] args) {
 // TODO Auto-generated method stub
 new BookImpl().bookName();
 }

}
```

ANS: code compiles and prints “In BookImpl Class”

Which of the below is not a valid classification of design pattern

Creational Pattern

Structural pattern

Behavioral Pattern

Java pattern

ANS: D Java pattern

### Section 1:

1. What is the output of the following code?

```

public class Main{
 public static void main(String args[]){
 int twoD[][] = new int [4][];
 twoD[0] = new int [1];
 twoD[1] = new int [2];
 twoD[2] = new int [3];
 twoD[3] = new int [4];
 for (int i=0;i<4;i++){
 for (int j=0;j<i+1;j++){
 twoD[i][j]; //Line 2
 }
 System.out.println("executed");
 }
}

```

**Ans – Compilation error in Line 2 as there is no Assignment operator used**

2. Which of the following statement is false about an object in java?

**Ans – Object can communicate with each other**

3. What is the output of the code given below?

```

public class ABC {
 public static void main(String args[])
 {
 boolean flag = false;
 if(flag = true)
 {
 System.out.println("true");
 }
 else
 {
 System.out.println("false");
 }
 }
}

```

**Ans- true**

4. Ria has a class called ‘Account.java’ under two separate packages. Com.infy.debit and com.infy.credit can see give the account.java of both the packages in another class called ‘ReportUtil.java’ of package com.infy.util?

**Ans – No She cannot as there will be compilation error stating the import collides with another import.**

5. Identify the suitable datatype to be used for variable ‘b’ at line 1?

```

class Expression {
 public void calc()
 {

```

```

 double x=10;
 int y = 20;
 float z= 30;
 //Line 1 --- Answer is double
 b=x+y+z;

 }

}

```

**Ans – double**

6. What is the result when the following code is compiled and executed?

```

class Demo {
 int x=1;
 int y=2;
 Demo display(Demo demoParam){
 Demo obj = new Demo();
 obj = demoParam;
 obj.x = demoParam.x++ + demoParam.y ;
 demoParam.y = demoParam.y;
 return obj;
 }
 public static void main(String[] args){
 Demo obj1 = new Demo();
 Demo obj2 = obj1.display(obj1);
 System.out.println("obj1.x = " + obj1.x + "obj1.y = " + obj1.y);
 System.out.println("obj2.x = "+ obj2.x + "obj1.y = " + obj2.y);

 }
}

```

**Ans:**

obj1.x = 4obj1.y = 3  
 obj2.x = 4obj1.y = 3

7. Have a look at the following class and predict the option that is correct?

```
public class CodeForException {
```

```

public void callMe() throws Exception{
 try{
 int value = 3/0;

 }catch (ArithmaticException ae){
 System.out.println(ae);
 }
 }
public void calling(){
 callMe();
}
}

```

**Ans:** The code will face issues during compilation as the calling code neither handles nor throws Exception

8. What will be output of the code when executed?

```

public class OperatorTester {
 public static void main(String[] args){
 int a = 10;
 int b = 15;
 if (++a <(b=b-=4) || (a=a+=4)>b++){
 System.out.println(a+", " +b);
 }
 }
}

```

**Ans – 15,12**

9. What is the result when the following code snippet is executed?

```

class Employee {
 static String name = " ";
 public static void main(String[] args) throws Exception{
 try{
 name+="jo";
 throw new Exception();
 }

 }catch(Exception e){
 name+="hn";

 }finally{
 name+="s";
 empName();
 name+="on";
 }
 }
}

```

```

 }
 System.out.println(name);
 }
 static void empName(){
 throw new NullPointerException();
 }
}

```

**Ans –** The code will give java.lang.NullPointerException in finally block when executing

- What is the result when the code is compiled and executed?

```

class Calculator {
 int a = 123, b=200;
 public void display(){
 System.out.println("a:" + a + "b:" + b+"");
 }
}
class CalculatorDemo{
 public static void main(String[] args)
 {
 Calculator calculator1 = new Calculator(); //Line1
 Calculator calculator2 = calculator1; //Line2
 calculator1.a+= 1;
 calculator1.b+= 1;
 System.out.println("calculator1 values:");
 calculator1.display();
 System.out.println("calculator2 values:");
 calculator1.display();
 }
}

```

**Ans –**

```

calculator1 values:
a:124b:201
calculator2 values:
a:124b:201

```

## Section2

- What can be expected when the following code is compiled

```

abstract class Customer {

 public int custId;//Line1
 Customer()//Line2
 {
 custId=23456;//Line3
 }
 abstract public void setId();
 abstract final public void getid();//Line4
}

```

**Ans : Option D: Compilation error in Line 4 abstract methods cant be final**

2. What is the correct way of placing ‘this’ keyword in a constructor?

**Ans Option A : First Statement**

3. At what position should Varargs be placed in a parameterised method

**Ans Option B: Last Place**

4. Predict the output of the following code

```

public class Demo {
 static int x=232;
 int y=135;
 public void display(){
 System.out.print("Inside Demo");
 }
 public static void show(){
 System.out.print(x);
 }

 public static void main(String[]args)
 {
 Demo.show(); //Line 1
 Demo demo=new Demo();
 demo.show(); //Line2
 show(); //Line3
 demo.display();
 }
}

```

**Ans: Option A: Compilation error in line2 as static method cannot be called with object reference**

5. Which of the following keyword can be used to restrict a class to be implemented in java

**Ans Option B. Final**

6. Given below. What will be the output when code is compiled and executed

```

public class Parent {

 public final void show(){
 System.out.println("show() inside Employee");
 }
}

final class Child extends Parent {
 public void show1() { //Line1
 final int x=100;
 System.out.println("show() inside Unit");
 System.out.println(x);
 }
}

public class Demo {

 public static void main(String[] args) {
 Parent parent = new Child();
 new Child().show1(); //Line2
 }
}

```

Ans: Option D

**show() inside Unit**

**100**

7. Which of the following is FALSE regarding parameterized constructor in Java

- a. parameterised constructor should have void as return type
- b. parameterised constructor can take any number of parameters
- c. parameterised constructor cannot have private access modifier
- d. parameterised constructor throw an exception

Ans Option c parameterised constructor cannot have private access modifier

8. What is the output of the following

```

public class Trainer{

 public void display(String name) {
 System.out.println("I am a trainer");
 print(name);
 }

 public void print(String name) {
 System.out.println(" I train "+ name+ ".");
 }
}

```

```

public class Trainee extends Trainer {
 String myname;

 public Trainee(String myname) {
 super();
 this.myname= myname;
 }

 public void display(String name) {
 super.display(name);
 System.out.println("I am a trainee");
 print("Java");
 }

 public void print(String name) {
 super.print(name);
 System.out.println("I want to learn " +name+"");
 }

 public static void main(String[] args) {
 Trainer trainee = new Trainee("XYZ");
 trainee.display("Java");

 }
}

```

Ans: Option B

```

I am a trainer
I train Java.
I want to learn Java
I am a trainee
I train Java.
I want to learn Java

```

9. Which of the following is FALSE regarding ‘this’ keyword in java?

Ans--- optionD, ‘this’ keyword can be used to create a block of code.

10. Given

```

Class Parent{}
Class Child extends Parent{}
Final class GrandChild extends Child{}

```

Which of the following statement is not ‘true’ about the above code.

Ans : Option C: Reference of parent class can accept the instance of child class but not the instance of GrandChild class

## Section 3

1. What will be the output of the below code

```
public class student {

 String stuName="Jacklin";

 void func() throws Exception{
 try
 {
 stuName+="--";

 }
 catch (Exception e) {
 throw new Exception();
 }
 finally
 {
 stuName+="Hello" +stuName;
 }
 stuName+="!!!!";
 }

 void disp() throws Exception
{
 func();
 System.out.println(stuName);
 }

 public static void main(String[] args) {
 try
 {
 student student = new student();
 student.disp();
 }
 catch (Exception e) {
 System.out.println("CatchBlock");
 }
 }
}
```

Ans Jacklin--HelloJacklin--!!!!  
Option D

2. Predict the output of the foll

```
public class Greeter{

 public static void main(String[] args) throws Exception{
 try
```

```

 {
 System.out.println("Greeting"+ ""+args[0]);

 }catch (ArrayIndexOutOfBoundsException e) {

 System.out.println("Sam");
 }

}

```

Ans : Option C Sam

3. Predict the output of the below code.

```

public class TestDemo{

 public static void main(String[] args) {
 for(int a=0;a<6;++a)
 try {
 if(a%3==0)
 throw new Exception("Except1");
 try {
 if(a%3==1)
 throw new Exception("Except2");
 System.out.println(a);

 }catch (Exception inside) {
 a +=2;
 }finally {
 ++a;
 }
 }catch (Exception outside) {
 a+=3;
 }finally {
 ++a;
 }
 }
 }
}

```

Ans: Option A: 5

4. Given

```
Public class Sample{
 Public static void main(String[] args) throws Exception{
 Try{
 System.out.println("In try block");
 System.exit(0);
 }catch(Exception ex){
 System.out.println("In catch block");
 Ex.printStackTrace();
 }finally{
 System.out.println("In finally block");
 }
 }
}
```

Predict the output?

Ans: Option B.In try block

5. Which of the following exceptions are ignored during compile time (choose any 2 option)

Option A and C

ArrayIndexOutOfBoundsException

NullPointerException

## Section 4

1. Identify the outcome of the given code snippet

```
public class Demo{

 public static void main(String[] args) {
 int [] arrVar = {11,22,33,44,55,66,77,88,99,109};
 int position =3;
 int value =7;
 for(int i= arrVar.length-1;i>position;i--) {
 arrVar[i]= arrVar[i-1];
 }
 arrVar[position]= value;
 System.out.println("New Array"+ Arrays.toString(arrVar));
 }
}
```

Ans :

Option D New Array[11, 22, 33, 7, 44, 55, 66, 77, 88, 99]

2.Which of the following are true about enums in Java? (Choose 3 correct)

- I.     enums can implement any other interface in java
- II.    an instance of enum can be created outside of enum itself

- III. enum cant extend any other class except the abstract base class java.lang.enum
- IV. enum can be used to implement singleton design pattern

Ans Options i, iii & iv

3. Predict the output of the below

```
public static void main(String[] args) {
 String name1 = "Java";
 String name2 = "Java";
 System.out.println(name1==name2);
 System.out.println(name1.equals(name2));
}
```

Ans Option A: True True

4. Given

```
public class App{

 public static void main(String[] args) {
 String s1 = new String ("smart");
 String s2 =s1;
 if(s1==s2) {
 System.out.println("==smart");
 }
 if (s1.equals(s2)) {
 System.out.println("equals smart");
 }
 }
}
```

Ans Option B

==smart  
equals smart

## Section 5

1.What will happen when the following code is executed? ( Output - not sure)

```
public class TestDemo{

 public static void main(String[] args){
 List list1=new ArrayList<>();
 list1.add("1");
 list1.add("2");
 list1.add(1, "3"); //Line1
 List list2=new LinkedList<>(list1); //Line2
 list1.addAll(list2); //Line3
 list2=list1.subList(2,6); //Line4
 list2.clear();
 System.out.print(list1+ "");
 }
}
```

Ans: [1, 3] (answer not in option)

2. Which among the following option are correct with respect to HashMap

- Override boolean equals(Object o)
- Override toString()
- Override int hashCode()
- Override String hashCode()

Ans – option a - Override boolean equals(Object o)

3. Identify the incorrect statement as per the collection Framework hierarchy?

Ans D: LinkedHashSet implements HashSet

4. What is the result of compiling and running this code snippet?

```
import java.util.Arrays;
import java.util.Comparator;

public class Demo3 implements Comparator<String>{
 @Override
 public int compare(String x, String y){
 return x.toLowerCase().compareTo(x.toLowerCase());
 }
 public static void main(String[] args){
 String[] values = {"JOHN", "Annie", "JACKLINE"};
 Arrays.sort(values,new Demo3());
 for (String str:values)
 System.out.print(str + " ");
 }
}
```

Ans - JOHN Annie JACKLINE

5. Predict the output of the below code snippet?

```
import java.util.ArrayList;
import java.util.List;

public class sam1 {
 public static void main (String[] args){
 List<String> list = new ArrayList();
 list.add(0,"A");
 list.add(1,"B");
 list.add(1,"C");
 for (Object object:list){
 System.out.print(" " + object);
 }
 }
}
```

Ans --- A C B

## Section 6:

1.What is true regarding the following code snippet

```
public interface StaticInterface {

 static void staticMethod()
 {
 System.out.println("Inside interface");
 }
}

public class StaticInterfaceImpl implements StaticInterface {

 public void staticMethod()
 {
 System.out.println("Inside class");
 }
}

public class StaticDemo {

 public static void main(String[] args)
 {
 new StaticInterfaceImpl().staticMethod();
 }
}
```

```
}
```

Ans: Option D. code will print “inside class” on execution

2. Which out of the following are true in regard to interfaces in Java
- a. An interface can contain default and static method with defined bodies
  - b. An object can be created of an interface
  - c. Multiple inheritance is allowed in interface
  - d. Multi level inheritance is not possible in interface

Ans Option A and C

3. Default format of LocalDate is

Ans Option B yyyy.mm.dd

4. Given

```
interface Greeting{
 default void greet() {
 System.out.println("In Greet interface");
 }
}

class GreetingDef implements Greeting{
 public void greet() {
 System.out.println("In GreetingDef class");
 }
}

public class App {

 public static void main(String str []){
 Greeting obj = new GreetingDef();
 obj.greet();
 }
}
```

Ans:Option B- In GreetingDef class

5. The below code will generate compilation error. Select the possible options to avoid it(Choose 2)

```
public interface Insurance{
```

```

static void policy() {
 System.out.println("policy");
}

public class InsuranceImpl implements Insurance{
 public void policyPayment() {
 policy();
 }
}

public class App{
 public static void main(String[] args) {
 new InsuranceImpl().policyPayment();
 }
}

```

- a. Static method of an interface can only be accessed using interfaces name
- b. Static method should always be public
- c. Static method cannot be invoked inside the non static method
- d. Policy() method of interface has to be accessed using interface name

**Ans option A & D**

## **Section 7:**

1. What is Magic Number in java in the context of java programming best practices?

**Ans – A direct usage of number in the code.**

2. Given

```
public class Employee {
 private int empId;
 private String empName;
 private String designation;
 private transient String dob;
}
```

Analyze the given code and identify the suitable comments from the below option.

Ans:

- (i) Fields in non-serializable classes should not be 'transient'
- (ii) Make the Employee class as serializable

3. From the below options identify the methods and constructors in Throwable that support checked exception?

- (i) Throwable getCause()
- (ii) Throwable initCause(Throwable)
- (iii) Throwable (String, Throwable)
- (iv) Throwable (Throwable)

Ans - (i), (iii), (iv)

4. Identify the valid code needs to be inserted in line5, assume the code is running in

```
import java.util.Random;
```

```
public class Emp {
 private static String id;
 private Random random = new Random();
 public Emp(){
 //line5 if(id==null){

 id = "ACC1101" + Math.abs(random.nextInt());
 }
 }
 public String getId(){
 return id;
 }
}
```

Ans - if(id==null){

5. Select the valid code fragment according to Java coding standard?

- (i) **public void draw(String s){**  
    **if(s.equals("Square")){**

```

 drawSquare();
 }
}
(ii) public void draw(String s){
 if("Square".equals(s)){
 drawSquare();
 }
}

```

**Ans:** Both (i) and (ii) are valid.

## Section 8:

1. Which among the following comes under Creational Design pattern?

**Ans:** Singleton Design Pattern

2. Which of the statements are true about design patterns?
  - i. There are only three design patterns defined for Java
  - ii. We can use each design pattern only once per application
  - iii. Design patterns are conceptual reusable solutions

**Ans:** Statement i,ii,iii

3. What changes need to be made in the following code to make the singleton pattern correct?  
(choose 2)

```

public class Employee1 {
 public static Employee1 employeeInstance;
 private Employee1(){}
 public static Employee1 getEmployee()
 {
 if (employeeInstance==null){
 employeeInstance = new Employee1();
 }
 return employeeInstance;
 }
}

```

**Ans:**

Option A &D

- A. None the singleton pattern is properly implemented
- D. Change the access modifier of employeeinstance from public to private

## Section 9:

1. Given

```

//Assume all the required imports are added
import java.util.ArrayList;

import org.junit.Before;
import org.junit.Test;

public class Demo4 {
 static int a = 10;
 static ArrayList b = new ArrayList();
 @Before
 public void init(){
 a=15;
 b.add(a);
 }
 @Test
 public void test(){
 a=a+20;
 System.out.print(a);
 System.out.println(b);
 }
 @Test
 public void test1(){
 a=a+30;
 System.out.print(a);
 System.out.println(b);
 }
}

```

Predict the output?

Ans -  
 35[15]  
 45[15, 15]

2. Which of the following annotation must be used in a test class to run same test again and again

Ans: @Test

3. Predict the output for the below

```

//Assume all the required import statements are added
import org.junit.Before;
import org.junit.Test;

import junit.framework.Assert;

public class TestDemo1 {
 @Before
 public void beforeTest1(){
 System.out.println("in before test2");
 }
 @Before

```

```

public void beforeTest2(){
 System.out.println("in before test1");
}
@Test
public void test(){
 String a = "123";
 Assert.assertEquals("123",a);
}

```

**Ans:**

Test Passed as they point the same object and prints the below in console  
 in before test1  
 in before test2

ThJava Question

1. Which of the following is not pre defined annotation in Java?

- @Deprecated
- @Overriden
- @SafeVarargs
- @FunctionInterface

**Ans: Option B(@Overriden)**

2. public class TestString3{  
 Public static void main(String[] args){  
 //insert code here//Line3  
 System.out.println(s)  
 }  
 }

Which of the below code fragment when inserted independently at line3 generate output as 498

**Ans Option B**

```
StringBuffer s = new StringBuffer("123456789").delete(0,3).replace(1,3,"98").delete(3,8);
```

3.Predict output of this

```

public static void main(String[] args) {

 String Value1= "Hello";

```

```

String Value2= new String ("Hello");
System.out.println(Value1.equals(Value2)+"+"+(Value1==Value2));
String Value3= Value2.intern();
System.out.println((Value1==Value3)+"+"+(Value1.equals(Value3)));
}

}

```

**Ans truefalse**  
**Truetrue**

66. Which of the following statements are true if a duplicate element obj T is added to a HashSet?

- a) The element obj T is not added and add() method returns false
- b) The element obj T is added successfully
- c) An exception occurs during runtime
- d) An exception occurs during compile time

. which of the following is incorrect regarding interfaces in Java SE8

- a.all the methods are public,abstract by default
- b.all the variables are public by default
- c.methods can have implementation
- d.its possible to hold static method

- a) a and b
- b) b and c**
- c) a,b and c
- d) a only

Which of the below are NOT good practices for creating objects?

- a) Lazy initialization of objects
- b) Creating String literals instead of String objects
- c) Creating Wrapper objects instead of primitives**
- d) invoking static factory methods for immutable classes

Which of the below statement indicate the need to use the factory pattern?

- a) we have two classes that do the same thing
- b) we only want one instance of the object to exist
- c) we want to build a chain of objects
- d) we don't want the caller to depend on a specific implementations**

4.What is the output when the below code is compiled and executed

```
Package exceptions;
Public class Demo
{
public static void main(String[] args) {

 try
 {
 return;

 }
 finally
 {
 System.out.println("Finally");
 }

}
```

**Ans : Option A:( Finally)**

What is the output of the below code snippet

```
enum Customer
{
private CUSTID
public CUSTNAME
protected ADDRESS

}
```

**Ans : D Compilation Fails**

(Explanation: Enum cannot have any modifiers. They are public, static and final by default)

What is the output of the below code

```
package javaBasics;
public class student {

 String stuName="Jackin";
 void display()
 {
 try
 {
 stuName+="John";
 func();
 }
 catch (Exception e) {
```

```

 stuName+="GoodName";
 }
 }

void func() throws Exception{
 try
 {
 stuName+=".....";
 method();
 }
 catch (Exception e) {
 throw new Exception();
 }
 finally
 {
 stuName+="!!!!!";
 }
 stuName+="hello";
}
void method() throws Exception
{
 throw new Exception();
}

void disp()
{
 System.out.println(stuName);
}

public static void main(String[] args) {
 try
 {
 student student = new student();
 student.display();
 student.disp();
 }
 catch (Exception e) {
 System.out.println("CatchBlock");
 }
}
} Ans Option D (JackinJohn.....!!!!GoodName)

```

What will be the output of the following code

```

public class Test{
 public void method()
 {
 for(int i=0;i<3;i++) {
 System.out.println(i);
 }
 System.out.println(i);
 }
}

```

```
 }
}
```

**Ans: C. Compilation Fails**

What is the result of attempting to compile and run this program

```
public class Customer{
 public static void main(String [] args)
 {
 Float f1= new Float(67.65f);
 Float f2= new Float(36.45f);
 if(f1>f2)
 {
 System.out.println("f1 is bigger than f2");
 }
 else
 {
 System.out.println("f1 is not bigger than f2");
 }
 }
}
```

**Ans : Option A (f1 is bigger than f2)**

What is the result of the following code snippet when compiled

```
public class Employee {
 int employeeid;
 double getEmployeeid()
 {
 System.out.println("Employee Id");
 return employeeid;
 }
}
```

Ans A: The code will not be complied as there is no main method

Given Enum definition and java class

```
enum Day{
```

```

 SUNDAY(1), MONDAY(2), TUESDAY(3), WEDNESDAY(4), THURSDAY(5),
FRIDAY(6) ,SATURDAY(7) ;
private int value;
private Day(int value)
{
 this.value =value;
}

public int getValue()
{
 return this.value;
}

}

public class Employee {
 public static void main(String[] args)
 {
 for (Day day: Day.values()) {

 //Line1
 System.out.print(day.toString()+"-");

 //System.out.print(day.name()+"-");

 }
 }
}

```

What should be [placed in line 1 to get the output as  
SUNDAY-MONDAY-TUESDAY-WEDNESDAY-THURSDAY-FRIDAY-SATURDAY-  
Choose one or more options

**Ans : A and C**

```

System.out.print(day.toString()+"-");
System.out.print(day.name()+"-");

```

What will be the output of the following code

```

public void method(){
 for(int i=0;i<3;i++){
 System.out.print(i);
 }
 System.out.print(i);
}

```

**Ans: C. Compilation fails**

which of the below exceptions are mostly thrown by JVM in a Java application?(Choose all that apply)  
means runtime exception

- a) ClassCastException
- b) IllegalStateException
- c) NumberFormatException
- d) IllegalArgumentException
- e) ExceptionInitializerError

Checked Exceptions are Compile time exceptions

Unchecked Exceptions are runtime exceptions

//Check Tutorial

What's the output of the following code

```
public class Demo {
 void main(){
 System.out.println("JAVA");
 }
 static void main(String args){
 System.out.println("Spring");
 }
 public static void main(String[] args){
 System.out.println("Hibernate");
 }
 void main(Object[] args){
 System.out.println("Apache Camel");
 }
}
```

**Ans Option A: Hibernate**

Given the below code snippet, predict the correct option

```
public class Operator {
 public static void main(String[] args){
 float val1=5.3f;
 float val2=2.3f;
 float result= val1 %val2;
 System.out.println(result);
```

```
 }
}
```

Ans: Option A **Code compiles, runs and produces the output 0.7000003**

What is the output for the below code

```
public class Employee {

 public final void show(){
 System.out.println("show() inside Employee");
 }

}

public class Dem011 {

 public static void main(String[] args) {
 Employee employee = new unit();
 new unit().show1();

 }
}

final class unit extends Employee {
 public void show1() {
 final int x=100;
 System.out.println("show() inside Unit");
 System.out.println(x);
 }
}
```

Ans:Option D.  
**show() inside Unit  
100**

What is the output when we execute the below code

```
public class Dem011 {
 static class Customer {
```

```

 public void go() {
 System.out.println("Inside Customer");
 }
 }
 static class Account extends Customer {
 public void go() {
 System.out.println("Inside Account");
 }
 }
 static class Branch extends Customer {
 @Override public void go() {
 System.out.println("Inside Branch");
 }
 }
 public static void main(String[] args) {
 Customer customer = new Account();
 Branch branch = (Branch) customer;
 branch.go();
 }
}

```

**Ans Option5:** An exception is thrown at runtime because (Branch)Customer is incorrect

Predict the output for the below code

```

public class Game {

 public static void main(String[] args) {
 displayRegistration("Hockey"); //Line 1
 displayRegistration("Kho-Kho", 132, 102, 36); //Line 2
 }

 public static void displayRegistration (String gameName, int... id) {
 System.out.println("Registration for " + gameName + ".");
 for(int i=0; i<id.length; i++) {
 System.out.println(id[i] + "");
 }
 }
}

```

**Ans**

```

Registration for Hockey.
Registration for Kho-Kho.
132
102
36

```

1. What is the output of the following code?

```
Class Employee {
 Void disp(char c){
 System.out.print("Employee name starts with : "+c+" .");
 System.out.print("His experience is : 11 years. ");
 }
}

Class Main extends Employee {

 Void disp(Char c) {
 Super.disp(c);
 System.out.print("Another employee name also starts with : "+c+" .");
 new Employee().disp("D");
 disp(7);
 }
}

String disp (int c) {
 System.out.print("His experience is :" +c+");
 return "Bye";
}
}

Public class Demo11 {
 Public static void main (String a[]){
 Employee emp = new Main();
 emp.disp("S");
 }
}
```

1. Employee name starts with : S. His experience is : 11 years. Another employee name also starts with : S. Employee name starts with : D. His experience is : 11 years. His experience is : 7.

2. Employee name starts with : S. His experience is : 11 years. Another employee name also starts with : S. His experience is 7 years
3. Employee name starts with : S. His experience is : 11 years. Another employee name also starts with : S. Employee name starts with : D. His experience is
4. Employee name starts with : S. His experience is : 11 years. Another employee name also starts with : S.

Predict the output of below code

```
public class Test {

 public static void main (String a[]){

 System.out.print("Implementing type Casting");
 Dog d = new Dog();
 BullDog bd = (BullDog) d;
 bd.show();
 }

}

class Dog{
 void show(){
System.out.print("Dog");
}
}
class Cat{
 void show(){
System.out.print("Cat");
}
}
class BullDog extends Dog{
 void show(){
System.out.print("BullDog");
}
}
```

**Ans tricky as runtime error and Implementing type casting also comes. But I guess more appropriate is runtime error**

**RUNTIME ERROR : java.lang. ClassCastException**

```
Implementing type CastingException in thread "main" java.lang.ClassCastException:
certificationJava.Dog cannot be cast to certificationJava.BullDog at
certificationJava.Dem011.main(Dem011.java:9)
```

Which code fragment can be inserted at Line 26 to display the output as “Inside catch(RuntimeException) finally end” ?

```
public class Dem011 {

 public static void main (String a[]){

 {
 try
 {
 method();
 System.out.print("Inside try");
 }
 catch (RuntimeException ex)
 {
 System.out.print("Inside catch(RuntimeException)");
 }
 catch (Exception ex1)
 {
 System.out.print("Inside catch(Exception)");
 }
 finally
 {
 System.out.print("finally");
 }
 System.out.print("end");
 }
 }
 public static void method()
 {
 //Line 26
 }
}
```

**Ans: Option A**

**throw new RuntimeException();**

Predict the output of the foll

Square.java

```
final class Square {
 private double length, breadth;
 Square(double length, double breadth) {
 this.length= length;
 this.breadth= breadth;
 }
 Square(Square square){
 System.out.println("Copy Constructor Invoked");
 length =square.length;
 breadth= square.breadth;
 }
 public String toString() {
 return "(" + length +"+"+breadth+ ")";
 }
}
```

Main.java

```
class Main{
 public static void main(String args[]){
 Square square1= new Square(110,115);
 Square square2= new Square(square1);
 System.out.println(square2);
 }
}
```

Ans: Option A

Copy Constructor Invoked  
(110.0+115.0)

Which Line fragment can be inserted in Line 1 will help to get the output as 110231(choose all apply)

```
package certificationJava;

public class InnerClassDemo {

 private int bookid=110;

 class Book
 {
 private int bookid=231;
 private int getBookid()
 {
 return bookid;
 }

 public void main (String [] args)
 {
 Book book = new Book();
 System.out.println(book.getBookid());
 }
 }

 private int getBookid() {

 return bookid;
 //Line1
 }
}
```

**Ans : Option 2 and Option 4**

```
InnerClassDemo innerClassDemo = new InnerClassDemo();
InnerClassDemo.Book book = innerClassDemo.new Book();
```

```
System.out.printf("%d",innerClassDemo.getBookid());
book.main(args);
```

and

```
InnerClassDemo innerClassDemo = new InnerClassDemo();
Book book = innerClassDemo.new Book();
System.out.printf("%d",innerClassDemo.getBookid());
book.main(args);
```

Which of the below option **fails** at Line 7(choose all that apply)

Employee.java

```
public class Employee {
 static final int empid =1101;
}

SuperDemo.class

class Unit extends Employee{
 int empid =1102;
 void display()
 {
 //Line7
 }
}
class SuperDemo {
 public static void main(String [] args)
 {
 Unit unit = new Unit();
 unit.display();
 }
}
```

## Options

- i. System.out.println("Maximum Speed"+super.empid);
- ii. System.out.println("Maximum Speed"+ new Employee().empid);
- iii. Employee emp1 = new Employee();  
System.out.println("Maximum Speed"+ new Unit().empid);
- iv. System.out.println("Maximum Speed"+ Employee.empid);

All 4 options works fine. Not sure if question is wrong or what

Given

The below code fragment can be inserted at Line 1 and Line 2.What will be the output?

```
ConstructorDemo1 constructorDemo1=new ConstructorDemo1(1101,"Jacklin");
ConstructorDemo1 constructorDemo2=new ConstructorDemo1(1102,"John",25);
```

```
class ConstructorDemo1 {
 private int id;
 private final String name;
 static final int age=22;
 ConstructorDemo1(int i,String n){
 id=i;
 name=n;
 }
 ConstructorDemo1(int i,String n,int a){
 id=i;
 name=n;
 }
 void display(){
 System.out.println(id+" "+name+" "+age);
 }
 public static void main(String args[]){
 //Line1
 //Line2
 constructorDemo1.display();
 constructorDemo2.display();
 }
}
```

```
}
```

**Ans Option B**

```
1101 Jacklin 22
1102 John 22
```

Predict the output of the below code

```
public class InnerClassDemo {
```

```
 InnerClassDemo()
 {
 System.out.print("InnerClassDemo Constructor");
 }
```

```
}
```

Demo.java

```
class Demo {

 Demo()
 {
 System.out.println("Demo Constructor");
 }
 public void disp()
 {
 System.out.print("Simple Class");
 }
}
```

```

public static void main(String[] args)
{
 InnerClassDemo innerClassDemo=new InnerClassDemo();
 innerClassDemo.createDemo();
}
void createDemo()
{
 (new Demo() {}).disp();
}

```

**Ans : Option A : Compilation fails**

Predict the output

```

class Main{

 public void display(int i) {
 System.out.println("Inside First");
 }
 public void method(int i,int j) {
 System.out.println("Inside Second");
 }
 public void method(int... k) {
 System.out.println("Inside Third");
 }

 public static void main(String args[]){
 new Main().method(110);
 }
}

```

```
 new Main().method(110,210);
 new Main().method(110,210,310); //Line1
 new Main().method(110110,210,310,410); //Line2

}
```

**Ans:**

```
Inside Third
Inside Second
Inside Third
Inside Third
```

**Predict the output**

```
public class Book {
 int bookid = 2356;
}

public class Book1 extends Book{
 int bookid = 1167;
}

public class Book2 extends Book1
{
 int bookid = 2378;
 void display()
```

```

 {
 System.out.println(super.super.bookid); //Line10
 System.out.println(super.bookid); //Line11
 System.out.println(bookid);
 }

}

class Demo {

 public static void main(String[] args)
 {
 Book2 book2 = new Book2();
 book2.display();
 }
}

```

**Ans: A. Compilation Fails because of an error in Line10**

When the following code is inserted in Line 6. Whats the output

```

Apple apple = (Apple)typeCastDemo;

class Apple {

}

public class TypeCastDemo {
 public static void main(String[] args)
 {
 TypeCastDemo typeCastDemo = new TypeCastDemo();
 }
}

```

```
//Line6
}
}
```

**Ans Option C.**

**Compilation fails as typecast cant be done from TypecastDemo to Apple**

What code fragment can be inserted at Line3 to enable the code to print 188.22

```
enum Fruits{
 APPLE,
 MANGO,
 STRAWBERRY,
```

```

LICHI;

double claculate(double a, double b) {
 switch(this) {
 case APPLE:
 return a+b;
 case MANGO:
 return a-b;
 case STRAWBERRY:
 return a*b;
 case LICHI:
 return a/b;
 default :
 throw new Assertion("Unknown input"+this);
 }
}

public class EnumDemo {
 public static void main(String[] args)
 {
 //Line3
 }
}

```

**Ans : Option A**

```
double res = Fruits.MANGO.claculate(298, 109.78);
```

Whats the output

```

public class Parent {

 void message()
 {
 System.out.println("Inside parent class");
 }
}

public class Derived extends Parent{

 void message()
 {
 System.out.println("Inside derived class");
 }

 void display()
 {
 message();
 super.message(); //Line1
 }
}

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

 Derived derived=new Derived();
 derived.display(); //Line2
 }
}

```

**Ans: Option D.**

**Inside derived class**

**Inside parent class**

What is the result of attempting to compile and run this program

```
public class Bank extends Exception{

}

public class Customer extends Bank {

}

public class ExceptionDemo {

 public static void main(String args[]) {

 try
 {
 throw new Customer();
 }

 catch (Bank bank) {

 System.out.println("Bank catch Block");
 }

 catch(Customer customer) {

 System.out.println("Customer catch Block");
 }
 }
}
```

**Ans : Option C. Compilation error because Customer class exception is not throwable.**

**Whats the output of the following**

```
public class Demo extends Book {
 int bookid =4567;

 public int getValue() {
 return bookid;
 }

 public void call() {
 System.out.println(getValue());
 System.out.println(super.getValue());
 }
}
```

```
public static void main(String args[]){
 Book book = new Book();
 book.call();//Line3
 super.call();//Line4
}
}
```

Book.java

```
public class Book {
 int bookid = 17897;

 public int getValue() {
 return bookid;
 }

}
```

**Ans: Option D. Compilation fails because of an error in line 3 and line 4**

2. What is the output of the following code ?

Package exceptions;

```
import java.io*;
Public class ExceptionDemo{
 Static class Car implements AutoClosable{
 Public void close(){
 System.out.print("Automatic Door Close");
 }
 }
 Static class carWindow implements Closerable{
 Public void close(){
 System.out.print("CarWindow");
 throw new RuntimeException();
 }
 }

 Public static void main(String[] args){
 Try(Car car=new Car();
 CarWindow carWindow=new CarWindow()){
 System.out.print("Inside try block");
 }
 Catch(Exception e){
 System.out.print("Inside catch block");
 }
 }
}
```

```

Finally{
 System.out.print("finally");
}
}
}
}

```

- a. Automatic Door close CarWindow Inside try block inside catch blockfinally
- b. Automatic Door Close CarWindow Inside catch blockfinally
- c. Inside try blockCarWindowAutomatic Door CloseInside catch blockfinally
- d. An exception is thrown at run time
- e. Compilation fails

**Ans : c. Inside try blockCarWindowAutomatic Door CloseInside catch blockfinally**

60. Given:

```

Public class ExceptionDemo1{
Static class Car implements AutoCloseable{
Public void close(){
System.out.print("Car door close");
Throw new RuntimeException();
}
}
Static class CarWindow implements Closeable{
Public void close(){
System.out.println("Car window close");
Throw new RuntimeException()
}
}
Public static void main(String[] args){
Try{
 //Line 1
}
Catch(Exception e){
System.out.println("Catch exception");
}
Finally{
System.out.print("finally");
}
}
}

```

Which of the below code can be inserted at Line1 to display THE OUTPUT AS “try block finally” (Choose all that apply)

A)Car car=new Car();  
**CarWindow carWindow=new CarWindow();**

**A) System.out.print("try block");**

**B) Car car=new Car();  
System.out.print("try block");**

**C) Car car=new CarWindow()  
System.out.print("try block");**

**D) system.out.print("try block")**

. Which two statements are true for a two-dimensional array?

- A.** It is implemented as an array of the specified element type
- B.** Using a row by column convention, each row of a two-dimensional array must be of same size
- C.** At declaration time, the number of elements of the array in each dimension must be specified
- D.** All the methods of the class Object may be invoked on the two-dimensional array

- a)** Option (A) and (B)
- b)** Option (A) and (B)
- c)** Option (B) and (C)
- d)** **Option (C) and (D)**

Which of the below code fragment needs to be inserted at Line12 to display the output as 15.

```
public class ExceptionInClass {
 int data=10 ;
 void calculate() throws Exception
 {
 try
 {
 data++;
 try
 {
 data++;
 // Line12
 }

 catch(Exception ex)
 {
 data++;
 }
 catch(Exception ex)
 {
```

```

 data++;
 }
 }

 void display()
 {
 System.out.println(data);
 }

 public static void main(String[] args) throws Exception
 {
 ExceptionInClass exceptionInClass = new
ExceptionInClass();
 exceptionInClass.calculate();
 exceptionInClass.display();
 }

}

```

**Ans: None of the above. If not write answer as option A**

What is the output when the below code is compiled and executed?

```

import java.util.regex.Matcher;
import java.util.regex.Pattern;

public class Demo1{
 public static void main(String[] args) {
 ("x*y");
 Matcher match=pattern.matcher("y");
 Boolean boolean1=match.matches();
 System.out.println(boolean1);
 }
}

```

**Ans: Option A True**

Output of below code is

```

class Light{
 Boolean isOn;

 void turnOn(){
 isOn=true;
 }

 void turnoff(){
 isOn=false;
 }

}

```

```

class LightDemo{
 public static void main(String[]args){
 Light light1=new Light();
 Light light2=new Light();
 light1.turnOn();
 System.out.println("light1 is on");
 light1.turnOff();
 System.out.println(light1.isOn);
 System.out.println(light2.isOn);
 }
}

```

**Ans : True**

**False**

**Null**

What will be the output of the code given below?

```

public class ABC{

 public static void main(String[]args){
 Boolean flag=false;
 if (flag = true){
 System.out.println("true");
 }
 if (flag = false)
 {
 System.out.println("false");
 }
 }
}

```

**Ans Option A. True**

What will be the output of the following code

```

public class Test{
 public void method()
 {
 for(int i=0;i<3;i++) {
 System.out.println(i);
 }
 }
 public static void main(String[]args){
 method();
 }
}

```

```
}
```

```
}
```

**Ans: C. Compilation Fails**

What error do we get when we compile the below code

```
public class Main{

 static int[] x;
 static{
 x[0]=102;
 }
 public static void main(String[]args){
 System.out.println(x);
 }

}
```

**Ans java.lang.ExceptionInInitializerError**

What is the output for the below code

```
public class VarArgsDemo{

 static void func(int...x)
 {
 System.out.println("Number of arguments "+x.length);

 for(int i:x)
 System.out.print(i+"");
 System.out.println();
 }
 void func(int a) //Line1
 { System.out.println("one"); }

 public static void main(String[]args){
 new VarArgsDemo().func(150);
 func(11, 12, 13, 14);
 func();
 }
}
```

**Ans:Option B**

one

Number of arguments 4

11121314

Number of arguments 0

How many objects are eligible for garbage collection after executing line8.

```
public class Employee {

 double salary;
 public static void main(String[]args){
```

```

Employee employee1=null;
Employee employee2=null;
employee1= new Employee();
employee2= new Employee();
employee1= new Employee();
Employee employee3=null;
employee1= employee3=null; //Line8
System.out.println("Hello World");
}
}

```

**Ans :3 objects**

Which statements are true about the following code snippet?(choose all that apply)

```

Public class Developer{}
Public class Employee{
Public String empName;
}

Public class Tester extends Employee{
Public Developer developer;
}

Public class Testing extends Tester{}


```

a. Testing has a empName  
b. Testing has a Developer  
c. Testing is a Developer  
d. Testing is a Employee  
e. Tester is a Testing  
f. Employee has a Developer

Ans : d & a

Observe the below code snippet:

```

public class Tree extends BasePlant{

 public void growFruit(){}
 public void dropLeaves(){}

}

public class BasePlant {

 private String name;
 public BasePlant(String name){
 this.name=name;
 }

 public String getName(){
 return name;
 }
}

```

Which of the following statement is true?(choose all that apply)

- a. The code will be compiled without any changes
- b. The code will be compiled only if the below code is added to the Tree class  
`Public Tree() {super("Plant");}`
- c. The code will be compiled only if the below code is added to the BasePlant class  
`Public BasePlant() {Tree(); }`
- d. The code will be compiled only if the below code is added to the BasePlant class  
`Public BasePlant() {this("Plant"); }`

Ans : b & d

Predict the output

Apple.java

```
public class Apple {
 public void color(){
 System.out.println("Red");
 }
}
```

Mango. Java

```
public class Mango extends Apple{

 @Override
 public void color(){
 System.out.println("Yellow");
 }
 public static void main(String[] args){
 Apple apple=new Mango(); //Line1
 apple.color(); //Line2
 }
}
```

**Ans: Yellow**

Given:

```
public interface interfaceDemo{
 //Line1
}
```

Select the suitable code fragment can be inserted at Line1(choose all that apply)

- a. `void display(int x);`
- b. `void display(int x) {}`
- c. `public static void display(int x){}`
- d. `default void display(int x);`
- e. `public interface Demo{}`

**Ans : a,c and e**

Analyze the code and select the suitable outcome

```
class Apple {
 private Apple() {
 System.out.println("Apple constructor");
 }

 void display(){
 System.out.println("Apple constructor");
 }
}

public class Main {

 public static void main(String[] args){
 Apple apple=new Apple(); //Line2
 apple.display();
 }
}
```

**Ans: Option D: Unresolved compilation problem: The constructor Apple() is not visible**

Output of the below code

```
public class Demo {
 static int x=232;
 int y=135;
 public void display(){
 System.out.print("Inside Demo");}
 public static void staticMethod(){
 System.out.print(x); //Line 8
 }

 public static void main(String[] args)
 {
 Demo.staticMethod(); //Line 13
 Demo demo=new Demo();
 demo.staticMethod(); //Line15
 staticMethod();
 demo.display(); //Line 16
 }
}
```

**Ans : Option B**

**232232232Inside Demo**

**Check line13. If its not Demo.staticMethod(); then answer will be 232232Inside Demo**

Output of the below code

```
public class Demo {

 public static void main(String[] args){
 try{
 throw 110;
 }

 catch(int ex){
 System.out.println("Caught Exception" + ex);
 }
 }
}
```

```
}
```

**Ans: Compilation Fails**

Predict the output of the below

```
public class TestDemo {

 public static void main(String[] args){
 int sum, a=10, b=10;
 try{
 System.out.println(sum=a/b);
 return; //Line 1
 } catch(ArithmetricException | Exception e){ //Line2
 System.out.println(e.getMessage());
 }finally{
 System.out.println("in finally");
 }
 }
}
```

**Ans: Compilation fails because of the error in Line2**

What are the different types of memory areas used by JVM(choose two)?

1.Class

**2.Heap**

**3.Stack**

4.Queue

JVM in java is a

1.Debugger

2.Assembler

**3.compiler**

4.Interpreter

132. What is magic number in java in the context of java programming best practices?

1.A number which gets printed on the console

**2.A direct usage of the number int the code**

3.A number which magically disappears from the code

4.A number which is generated through error

What is the output of the below code

```
public class person {

 public person(String name){
 System.out.println(name);
 }

}

public class student extends person{

 public student(){ //Line 8
 System.out.println(" Student");
 }

 public static void main(String[] args) { // Line 11
 new person("Jacklin");
 }
}
```

**Ans: Compilation fails because of an error in Line 8**

3. Given:

```

public abstract class Employee {
 private int empld;
 private int salary;
 public abstract void display();
 public void setValues(int empld, int salary){
 this.empld = empld;
 this.salary = salary;
 }
}

```

Which of the following classes provide the right representation of the child class of Employee class?

- 1) public abstract class Child extends Employee {  
    private int z;  
}
- 2) public class Child implements Employee {  
    private int z;  
}
- 3) public class Child extends Employee {  
    private int z;  
    public void display();  
}
- 4) public class Child extends Employee {  
    private int z;  
    public void display() {  
        /\* code here \*/  
    }  
}

Ans : 4) public class Child extends Employee {

```

 private int z;
 public void display() {
 /* code here */
 }

```

4. Given an abstract Class Customer as below:

```

public abstract class Customer
{

```

```
public abstract String getCustomerType();
}
```

Select a Valid implementation of getCustomer Type method in another class, from the below options:

- 1) abstract class C1 extends Customer{  
 public string getCustomer Type()  
 { return "Premium";  
 }  
}
- 2) Customer customer = new Customer(){  
 public String getCustomerType()  
 { return "Premium";  
 }  
}
- 3) class C1 extends Customer{  
 public String getCustomerType()  
 { return "Premium";  
 }  
}
- 4) new Customer(){  
 public String getCustomerType()  
 { return "Premium";  
 }  
}

Ans : 3) class C1 extends Customer{

```
public String getCustomerType()
{ return "Premium";
}
}
```

Output of the following

```
class Customer{
int customerId = 11201;
Customer() {
 customerId = 11240;
}
```

```
}

class Main {
public static void main(String args[]){
 Customer customer = new Customer();
 System.out.println(customer.customerId);
}
}
```

Ans: Option B 11240

Code to be written to get the output as below

False

Simple

Demo

For

Regular

Expressions

Using

Pattern

Matching

```

public class RegExDemo {
 public static final String string1="Simple demo for +"regular expressions"+"
using pattern matching";
 public static void main(String[] args){
 //Line 1
 //Line2
 }
}

```

**Ans: Option 1**

```

System.out.println(string1.matches("\t"));
String[] splitString=(string1.split(" " + ""));
//(String1.split(\s+)) not working in my computer so did like this
for(String string: splitString){
 System.out.println(string);
}
System.out.println(string1.replaceAll("\S"," \t"));
}

```

What is the result of attempting to complete and run this program?

```

public class Demo {
 public static void main(String[] args){
 String c="a";//Line 3
 switch(c) {
 //Line4
 case 65://Line5
 System.out.println("One");
 break;
 case "a"://Line6
 System.out.println("two");
 case 3://line 10
 System.out.println("three");
 break;
 }
}

```

```
 }
}
```

**Ans:D Computation fails because of an error in Line 5 and Line 10**

Select all possible options that are valid among the following Enums can be defined inside\_\_\_\_\_

- a) An interface
- b) A class {Multiple choice question}
- c) A static Context
- d) A method

Which code fragment can be inserted at Line 1 to enable the code to print as “Number of Days =25”

```
class Demo {
```

```

public static void main(String[] args)
{
 int monthValue=2;
 int yearValue=4000;
 int numberOfDays=10;

 switch(monthValue) {

 case 1: case 3: case 5:
 case 7: case 8: case 10:
 case 12;
 numberOfDays=31;
 break;
 case 4: case 6:
 case 9: case 11:
 numberOfDays=28;
 break;
 case 2:
 //Line1

 numberOfDays=25;
 else
 numberOfDays=28;
 break;

 default:
 System.out.println("Number of Days =" +numberOfDays);

 }
}

```

**Ans: Option not clear. Either one can come. Make sure assignment operator is there(= =)**

```

if((yearValue% 4 ==0) &&
 (yearValue% 100==0)
 ||(yearValue% 400==0))

```

**Or**

```

if((yearValue% 4 ==0) ||
 (yearValue% 100==0)
 ||(yearValue% 400==0))

```

81. Identify which of the following class breaks its input into tokens using a whitespace pattern?

- a. InputStreamReader
- b. Console

- c. Scanner
- d. Buffered Reader
- e. DataInputStream

What is the output when below code is compiled and executed.

```
import java.util.regex.Pattern;
public class RegExDemo2 {
 private static final String String1="";
 private static final String String2="one two three four five";
 public static void main (String[] args){
 Pattern pattern=Pattern.compile(String1); //Line 7
 String[] strArr=pattern.split(String2); //Line 8
 for(String str:strArr){
 System.out.println(str);
 }
 }
}
```

**Ans : Option C (if pattern.compile and pattern.spilt don't have dot means then compilation error)**

C) one  
two  
three  
four  
five

What is the output of the below code

```
package certificationJava;

public class ABC {
 public static void main(String args[]){
 Boolean flag=false;
 if(flag==true){
 System.out.println("true");
 }
 if(flag==false){
 System.out.println("false");
 }
 }
}
```

**Ans: Option A . True**

Which is the correct code fragment to be inserted at Line 1 to execute the code to print count starts from 111,112,113....

```
public class Demo2 {
```

```

 public static void main(String[] args){
 int[]X={111,112,113,114,115,116,117,118,119,110};
 //Line1
 System.out.println("count is"+i);
}
}

```

**Ans : Option B**

```
for(int i:X){
```

Predict the output of the below

```

public class Calculator {

 int a=123;
 int b=200;
 public void display(){
 System.out.println("a"+a+"b"+b+"");
 }
}

public class CalculatorDemo {
public static void main(String[] args)
{
 Calculator calculator1=new Calculator(); //Line1
 Calculator calculator2= Calculator1//Line2
 calculator1.a+=1;
 calculator1.b+=1;
 System.out.println("calculator1 values");
 calculator1.display();
 System.out.println("calculator2 values");
 calculator2.display();
} }
```

**Ans : D. Compilation fails because of error in Line2**

Output of the following

```

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

 int i1=0;
 int[] j={11,111,14,19,116,215}; //line4
 for (int i1:j) //line5
 System.out.printf("%d",i1);
 }
}
```

**Ans :Option C: compilation fail because of an error in line5**

Output of the following

```
abstract class Customer {

 public int custId;
 Customer()
 {
 custId=23456;
 }
 abstract public void setId();
 abstract final public void getid(); //Line11
}

class Demo extends Customer{
 public void setId(int custId)
 {
 this.custId=custId;
 }
 final public void getid() //Line9
 {
 System.out.println("Customerid"+custId);
 }
 public static void main(String[] args)
 {
 Demo demo=new Demo();
 demo.setId(1102);
 demo.getid();
 }
}
```

Ans :

- a) compilation fails because of an error in Line9
- b) compilation fails because of an error in Line11

Output of the below code

```
public class Employee {

 public final void show(){
 System.out.println("show()inside Employee");
 }
}

final class Unit extends Employee{
 public void show1(){ //Line1
 final int x=100;
 System.out.println("show()inside Unit");
 System.out.println(x);
 }
}

public class Demo11{
 public static void main(String[] args){
 Employee employee=new Unit();
 new Unit().show1();
 }
}
```

**Ans:Option D**

**show()inside Unit  
100**

Given

```
Class Parent{
}
Class Child extends Parent{
}
Final class GrandChild extends Child{
}
```

Which of the following statement is not true about the above code?

- a) The above code represents the multi-level inheritance with the two level
- b) The GrandChild class can Access the protected and public members of the parent and child class
- c) **Instance of parent class can accept the reference of the child class but not the reference of GrandChild class**
- d) The GrandChild class can override the methods of both Parent class and Child class

In the below code snippet identify which of the following method compares the given values and return an int which tells lesser or greater

```
public class WrapperClassDemo {

 public static void main(String aa[]){
 int x=90;
 Integer i1=new Integer(x);
 int y=90;
 Integer i2=new Integer(y);

 System.out.print(i1.compareTo(i2)+" "+Integer.compare(i2,i1)+" "+i1.equals(i2)+" "+
 (i1==i2));
 }
}
```

- a) **Compare()**
- b) Equals()
- c) compareTo()
- d) ==

Output of the below code

```
public class TestDemo {
 public static void main(String[] args){
 Integer n1=new Integer(100);
 Integer n2=new Integer(100);
 Integer n3=127;
 Integer n4=127;
 Integer n5=128;
 Integer n6=128;
 int n7=129;
 int n8=129;
 System.out.print(n1==n2);
 System.out.print(n3==n4);
 System.out.print(n5==n6);
 System.out.print(n7==n8);
 }
}
```

**Ans A: false true false false true**

Output of the following

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

 TreeSet<String> treeset=new TreeSet<String>();
 treeset.add("first");
 treeset.add("First");
 treeset.add("Second");
 System.out.println(treeset.ceiling("Fir"));
}
```

**Ans : Option C**

**First**

**If <String> not defined then compilation error comes**

Output of the following

```
public class TestDemo {
```

```

public static void main(String[] args){
ArrayList Strings=new ArrayList();
Strings.add("aAaA");
Strings.add("AaA");
Strings.add("aAa");
Strings.add("AAaa");
Collections.sort(Strings);
for(string:Strings){
System.out.print(string);
}}}

```

**Ans: Option A Compilation Fails**

Output of the following

```

public class person {
 private final String name;

 public person(String name){
 this.name=name;
 }

 public String toString(){
 return name;
 }
}

import java.util.TreeSet;
public class Group extends TreeSet {

 public static void main(String[] args){
 Group g=new Group();
 g.add(new person("Hans"));
 g.add(new person("Jane"));
 g.add(new person("Hans"));
 System.out.println("Total"+g.size());
 }

 public boolean add(Object o){
 System.out.println("Adding"+o);
 return super.add(o);
 }
}

```

**Ans : Option A**

a) Adding Hans

An exception is thrown at the runtime(`java.lang.ClassCastException`)

Output of the following

```
public interface StaticInterface {
```

```

static void staticMethod()
{
System.out.println("Inside interface");
}
}

public class StaticInterfaceImpl implements StaticInterface {

public void staticMethod()
{
System.out.println("Inside class");
}
}

public class StaticDemo {

public static void main(String[] args)
{
new StaticInterfaceImpl().staticMethod();
}
}

```

**Ans: Option D. code will print “inside class” on execution**

**Inside class**

Output of the following

```

import java.time.LocalDate;
import java.time.format.DateTimeFormatter;

public class Formatting {
 public static void main(String[] args)
 {
 LocalDate date=LocalDate.of(2016,11,13);

 DateTimeFormatter formatter= DateTimeFormatter.ofPattern("dd/MMM/YYYY");
 System.out.println(date.format(formatter));
 }
}

```

**Ans: Option D: 13/Nov/2016 will be printed**

**13/Nov/2016**

Predict the output

```
public interface Interface1 {
```

```

 default void method1()
 {
 System.out.println("Inside default method");
 }

}

public interface DefaultExtends extends Interface1{
 default void method1()
 {
 System.out.println("Default method redefined");
 }
}

public class InterfaceWithDefaultMethod implements DefaultExtends{
 public static void main(String[] args)
 {
 InterfaceWithDefaultMethod defaultExtend=new
 InterfaceWithDefaultMethod(); //Line4
 defaultExtend.method1(); //Line5
 }
}

```

**Ans: Option B**

**Default method redefined**

What happens when default keyword is removed from the below code snippet

```

public interface Interface1 {

 default void method1()
 {
 System.out.println("Inside default method");
 }
}

```

a.method cannot be overridden in the implementing classes

b.method can be overridden in the implementing classes

c.method cannot be given body in the interface

d.compilation error occurs

- a) a and b
- b) a,b and c
- c) c and d**
- d) b and c

Select the valid code fragment according to java coding standard?

- 1) public void draw(String s){

```

if(s.equals("Square")){
 drawSquare();
}
if(s.equals("Rectangle")){
 drawRectangle();
}
}

2) public void draw(String s){
 if("Square".equals(s)){
 drawSquare()
 }
 if("Rectangle".equals(s)){
 drawRectangle();
 }
}

```

**only option(1) is valid**

only option(2) is valid

Both(1) and (2) are valid

Both(1) and (2) are invalid

Whats the output of the below

```

public class Ex1 {
 public String formatInInput(String i){
 if(i.trim().length()==9){
 StringBuilder s1=new StringBuilder();
 s1=s1.insert(0, "+1(");
 s1=s1.insert(6, ")");
 s1=s1.insert(10, "-");
 return s1.toString();
 }
 return null;
 }
 public static void main(String args[]){
 Ex1 ob=new Ex1();
 String I;
 ob.formatInput(I);
 }
}

```

- a) compilation fails at Line3
- b) Compilation fails at Line 6
- c) Null pointer exception will be thrown if the value of I is null**

- d) Compilation fails due to error in Line7

```
// I got below compilation error
```

```
Exception in thread "main" java.lang.Error: Unresolved compilation problem:
```

Consider the below code snippet

```
Locate locate=new Locate("da","DK");
NumberFormat numberFormat=NumberFormat.getInstance(locate);
String number=numberFormat.format(100,99);
System.out.println(number);
```

Here NumberFormat.getInstance() follows which design pattern?

- a) Factory method pattern
- b) Singleton pattern
- c) Abstract Factory pattern
- d) Builder pattern

Output of the following

```
public class TestDemo {

 static int a=0;
 static ArrayList b;

 @BeforeClass
 public static void beforeClass(){
 a=10;
 b=new ArrayList();
 }
 @BeforeMethod
 public void int1(){
 a=15;
 b.add(a);
 }
 @Test
 public void test(){}
```

```

 a=a+20;
 System.out.print(a);
 System.out.println(b);
}
@Test
public void test1(){
 a=a+30;
 System.out.print(a);
 System.out.print(b);
}
}

```

Predict the output?

- a) 35[15]  
45[15,15]
- b) 35[15]  
65[15,15]
- c) 35[15]  
45[15]
- d) 35[15]  
65[15]
- e) 35[15]  
65[30]

Output of the below code snippet

```

public class Test {

 public static void main(String[] args){
 int [][] x;
 x=new int[3][4];
 for(int i=0;i<3;i+=2){
 for(int j=0;j<4;j++){
 x[i][j]=i+j;
 System.out.print(x[i][j] + " ");
 }
 }
 }
}

```

**Ans : Option B**

**01232345**

Output of the foll

```
public class Pet {
 public void displayName(){
 System.out.println("Inside Pet");
 }
}

public class Dog extends Pet{

 public void displayName(){
 System.out.println("Inside Dog");
 }
}

class Demo{
 public static void main(String[] args){
 Pet pet=new Dog();
 pet.displayName();
 }
}
```

**Ans Option D: Compilation Fails.**

Output of the following

```
public class Hello {
 public static void main(String[] args){
 String s="How\"are\"you?";
 System.out.println(s);
 }
}
```

**Ans: Option A**

**How"are"you?**

Output of the following

```
public class WrapperClassDemo {
```

```

public static void main(String args[]) {

 Integer intWrapper=Integer.valueOf("12345");
 Integer intWrapper2=Integer.valueOf("11",2);
 Integer intWrapper3=Integer.valueOf("E",16);
 System.out.println(intWrapper+" "+intWrapper2+" "+intWrapper3);
}

```

**Ans: Option C**

**12345 3 14**

Predict the output of the following

```

public class Demo11{
 public static void main(String args[]) {
 Set numbers=new HashSet();
 numbers.add(new Integer(45));
 numbers.add(88);
 numbers.add(new Integer(77));
 numbers.add(null);
 numbers.add(789L);
 Iterator iterator=numbers.iterator();
 while(iterator.hasNext())
 System.out.print(iterator.next());
 }
}

```

**Ans: Option F**

**null1789884577**

Which of the below code has to be inserted at Line1, to sort the keys in the props HashMap variable?

```

public class Demo11{
 public static void main(String args[]) {
 HashMap props=new HashMap();
 props.put("key45","some value");
 props.put("key12","some other value");
 props.put("key39","yet another value");
 Set s=props.keySet();
 //Line1
 }
}

```

**Ans: Option B. Collections.sort(s);**

Output of the following

```
public static Collection get(){
 Collection sorted=new LinkedList();
 sorted.add("B");
 sorted.add("C");
 sorted.add("A");
 return sorted;
}
public static void main(String[] args){
 for(Object obj: get()){
 System.out.print(obj+".");
 }
}
```

**Ans : Option B  
B.C.A.**

Output of the following

```
public static void main(String[] args){
 TreeSet tset=new TreeSet();
 tset.add(new item());
 TreeSet b=tset;
}
```

**Ans : Option A. Compilation Fails.**

Which of the following code snippet can be inserted at line1 to display the output as

76  
Hello

```

class Apple<A> {
 A obj;
 Apple(A obj)
 {this.obj=obj;
 }

 public A getObject()
 {return this.obj;

 }
}

class Main{
 public static void main(String[] args){
 //Line1
 }
}

```

**Ans :Option A and Option B**

```

Apple apple=new Apple(76);
System.out.println(apple.getObject());
Apple appleObj=new Apple("Hello");
System.out.println(appleObj.getObject());

```

82. Refer the below code snippets and predict the outcome?

```

Public class RepeatingAnnotations{
 @Retention(RetentionPolicy.RUNTIME)
 public @interface Chocolates{
 Favourite[] value() default();
 }

 @Favourite("Diary Milk")
 @Favourite("Kit Kat")
 @Favourite("5 star")
 @Favourite("Galaxy")
 public interface Chocolate{
 }

 @Repeatable(value=Chocolates class)
 Public @interface Favourite{
 String value();
 }

 Public static void main(String[] args){
 Favourite[] a=Chocolate.class.getAnnotationsByType(Favourite.class);
 }
}

```

```
Chocolates chocolates=Chocolate.class.getAnnotation(Chocolates.class); //Line5
for(Favourite favourite: chocolates.value()){
 System.out.println(favourite.value()); } }
```

- a. Nothing will be displayed
- b. null will be printed
- c. Runtime exception will be thrown at Line 5
- d. Dairy Milk

Kit Kat

5 Star

Galaxy

Output for the following

```
public class RepeatingAnnotations {
 @SuppressWarnings("all") //line1
 @SuppressWarnings("deprecation") //line2
 public void over()
 {
 new Date().setDate(00); }
 }
```

**Ans: Option B.**

**Compilation will not be successful as @SuppressWarnings annotation is non-repeatable in nature**

Output of the following

```
public class TestDemo {
 public static void main(String[] args){
 LocalDateTime date1=LocalDateTime.of(2017, Month.FEBRUARY,
11, 15, 30); //Line1
 LocalDateTime date2=LocalDateTime.of(2017, 2, 12, 10, 20);

 System.out.println(date1.compareTo(date2));
 }
}
```

**Ans Option A -1 will be printed as execution result**

Predict the output

```
class Apple {
 int quantity;
}

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

 Apple apple;
 System.out.println("apple quantity");

 }
}
```

**Ans: Option 5: Apple Quantity**

Output of the following

```
public class TestDemo {
 private static Object staticObject;
 public static Object createStaticObject(){
 if(staticObject==null){
 staticObject=new Object(0);
 }
 return staticObject;
 }
}
```

What changes are required in the above code for successful execution?

- 1.The method createStaticObject should be synchronized
- 2.The method createStaticObject should be private
- 3.The staticObject reference should not be static
- 4.The method createStaticObject should not return Object type

Output of the following

```
public class TestDemo {

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

```

LocalDate date=LocalDate.of(12,11,2017);
System.out.print(date);
}
}

```

Ans: Option D: Exception will be raised as date not in range

Output of the following

```

public class Demo
{
 public void division(int x,int y){
 try{
 int z=x/y;
 }
 catch(Exception e){
 System.out.print("Arithmetic Exception");
 }
 finally{
 System.out.print("finally block");
 }
 }
 public static void main(String[] args)
 {
 Demo demo=new Demo();
 demo.division(0,8);
 }
}

```

**Ans: Option 2. finally block**

Output of the below code

```

public class Demo
{
 void display()
 {
 System.out.println("x= "+x+" y= "+y);
 }

 public static void main(String[] args)
 {
 Demo thisDemo=new Demo();
 thisDemo.get().display();
 }
}

```

**Ans: C . Compilation Fails**

Output of the following

```
public class TestDemo {
 static void myCode() throws MyException{
 try{
 throw new MyException("Test exception");
 }
 catch(Error|Exception ex){
 System.out.print("Inside Error and Exception");
 }
 }

 public static void main(String[]args) throws MyException{
 try{
 myCode();
 }
 catch(Exception ex){
 System.out.print("Inside Exception");
 }
 }
}
```

**Ans Option A**

**Prints Inside Error and Exception**

Output of the following

```
public class ThisDemo
{

 int x;
 int y;
 ThisDemo(){
 x=45;
 y=56;
 }
 ThisDemo get() //Line1
 {
 return this;
 }

 void display()
 {
 System.out.printf("x=%d+y=%d",x,y);
 }

 public static void main(String[] args)
 {
 ThisDemo thisDemo=new ThisDemo();
 thisDemo.get().display();
 }
}
```

Ans: will know answer based on syso only

I got answer as  $x=^*+x+^*y=^*+y$

```
public class student {

 private School school;
 private StudentDetails stuDetails;
 private Fees fees;

 public MarksHistory marksHistory(Marks marksDetails){
 //computation
 }
}
```

**Ans: Lazy Initialization**

Output of the following

```
public class Demo11{
 public static void main(String[] args){
 Parent obj =new Child();
 obj.display();
 }
}

public class Parent {

 public void display(int a){
 System.out.println("Parent Method");
 }
}

public class Child extends Parent {

 public void display()
 { System.out.println("Child Method");
 }
}
```

**Ans: A: Compilation Fails**

Predict the output

```
public class Manager extends Employee {

 public void someManagerMethod(){
```

```

 //...
 }

public class Officer extends Employee {
{
 //...
 public void someMethod(Employee e){
 Manager m=(Employee)e ; //Line 12
 m.someManagerMethod();
 }
}

}

public class Demo {

 public static void main(String s){
 Officer obj=new Officer();
 obj.someMethod(new Officer()); //Line 19
 }
}

```

**Ans: Option 1: Compilation fails because of an error in Line 12**

Output of the following

```

public interface Demo1 {
 public void display(String points);

}

public class Demo2 implements Demo1{
 public void display(String points){};

}

public class Demo3 {
 public Demo1 disp(){
 return null; //more code here
 }
 public String displayValue(){ //Line6
 return null;
 //more code here
 }
}

public class Demo4 extends Demo3{
 public Demo2 disp(){
 //more code here
 return null;
 }
 private String displayValue(){
 //more code here
 }
}

```

```
}
```

Ans: Option C. compilation of class Demo4 will fail because of an error in line6

Which of the code segment is written using best practice

1. 

```
List list;
public List getList{
 if(list.size()==0)
 return null;
 else
 return list;
}
```
2. 

```
Integer i1=new Integer(11);
Integer i2=new Integer(11);
System.out.println(i1==i2);
```
3. 

```
String[] str=new String[]{"Hi","Hello","Welcome"};
List strList=Arrays.asList(str);
for(iterator itr=strList.iterator();itr.hasNext();){
 System.out.println(itr.next());
}
```

Ans : Code 2 only is valid

144. //Assume that the first two of three test cases fail in “Testclass”

// Assme all the required import statements are added

```
Public class testrunner{
 Public static void main(String [] args){
 Result result = junitcore.runclasses(testclass.class)
```

```

For (Failure failure : result.getfailures()){
 System.out.println(result.wassuccessful());
}

}

1) False
2) True
3) False false true
4) False false false

```

Output of the foll

```

public class collectionsDemo{
 public static void main(String argv[]){
 ArrayList arrList=new ArrayList();
 ArrayList arrListStr=arrList;
 ArrayList arrListBuf=arrList;
 arrListStr.add(1, "SimpleString");//line6
 StringBuffer strBuff=arrListBuf.get(0);//line7
 System.out.println(strBuff.toString());//line8
 }
}

```

**Ans: Option C.Compilation fails because of error in Line7**

Output of the following

```

public class StringTest {

 public static void main(String[] args){
 String joinString=String.join(".", "java", "programming", "course");
 String s1="JAVA", s2="java", s3="Java";
 s1.toLowerCase();
 s3=s3.replace("J", "j");
 System.out.println(joinString);
 System.out.println(s1.equals(s2)+" "+(s2==s3));
 }
}

```

**Ans Option D:**

**java.programming.course  
false, false**

### **Output of following**

```
public interface DefaultMethodInterface1 {
 default public void defaultMethod(){
 System.out.println("DefaultMethodInterface1");
 }

public interface DefaultMethodInterafce2 {
 default public void defaultMethod(){
 System.out.println("DefaultMethodInterface2");
 }

public class TestDemo implements DefaultMethodInterface1, DefaultMethodInterafce2{

 public static void main(String[] args){
 DefaultMethodInterface1 defMethln=new TestDemo();
 defMethln.defaultMethod();
 }
}
```

**Ans: Compilation fails**

. Which of these statements compile?(choose at least one)

checkbox

1. HashSet hs=new HashSet();

2. HashSet set=new HashSet();

```
3.List list=new Vector();
List values=new HasgSet();
List objects=new ArrayList();
Map hm=new HashMap();
```

#### Output of the foll

```
public class TestDemo {
 public static void main(String[] args){
 List list1=new ArrayList();
 list1.add("1");
 list1.add("2");
 list1.add("3");
 List list2=new LinkedList(list1);
 list1.add(list2);
 list2=list1.subList(2,5);
 list2.clear();
 System.out.print(list1+"");
 }
}
```

**Ans: Option 1**

**the program complies successfully and throws exception during runtime**

## Section 1:

1. Which of the following OOP terminology associated with java..... Employee has address

Ans - Inheritance

2. What is the result when the following code is compiled and executed

```
public class Test {
 Long a; //Line1
 long b;
 public Test(long c){
 b=a+c; //Line 2
 System.out.println(b);
 }
 public static void main(String[] args){
 new Test(new Long(10L));
 }
}
```

Ans: Null pointer exception in Line 2 as variable a is not

3. Given

```
class Movie implements Comparator<Integer> {
 public int comparator(Integer o1, Integer o2){
 return o2.compareTo(o1);
 }
 @Override
 public int compare(Integer o1, Integer o2) {
 // TODO Auto-generated method stub
 return 0;
 }
}
```

```

class MovieApp {
 public static void main(String[] args){
 Integer mov[] = {2019,2017,1989,1994};
 Arrays.sort(mov,new Movie());
 for (int i:mov){
 System.out.print(i+" ");
 }
 }
}

```

Ans - c

2019 2017 1989 1994

4. Identify the output:

```

public class MyDemo {
 public static void main(String[] args){
 int i =5;
 switch(i){
 case 1:
 System.out.println("One");
 break;
 case 2:
 //Line 1
 case 3:
 //Line 2
 System.out.println("Two and Three");
 case 4,5:
 //Line3
 System.out.println("Four and Five");
 break;
 default:
 System.out.println("Default");
 }
 }
}

```

Ans: Compilation error in Line 3 as multiple values are not allowed in case

5. Which of the following is correct usage of a relational operator made in if statement.

Ans : if (firstName.equals("Annie")&&salary==50000)

6. Identify the output of the below code:

```
public class TestDemo {
 public static void main(String[] args){
 boolean a = true;
 boolean b = true;
 boolean c = false;
 boolean d = true;
 System.out.println(a&&b || c&&d);
 }
}
```

Ans - true

7.

```
public class UtilTest {
 @Rule
 public ExpectedException thrown = ExpectedException.none();
 //@Test(expected = Exception.class)
 //Line1
 @Test
 public void test1() throws Exception{
 thrown.expect(NullPointerException.class);
 throw new NullPointerException();
 }
}
```

Ans - @Test

8. Which of the following component is responsible to compile, debug a java program?

Ans-JDK

9. What is the output for the below code?

```
interface Fruits{
 public void printPrice();
}
public class Apple {
 public static void main(String[] args){
 Fruits fruits = new Fruits(){
 public void printPrice(){
```

```
 System.out.println("150");
 }

};

fruits.printPrice();
}
}
```

Ans-150

10. Which among the following is valid option for wildcards?(select 2 options)

- A. Used to relax restriction on the variable
- B. Used in scenario where type being operated upon is not known
- C. Used in generic method type argument
- D. Can access members of super class

Ans:

- A. Used to relax restriction on the variable
- B. Used in scenario where type being operated upon is not known

11. Which of the below method name is valid as per Java naming convention?

Ans: methodName

12. Consider the Junit test class with junit fixture annotations and the methods as below:

```
@BeforeClass ---- init()
@BeforeClass ---- close()
@Before ---- setUp()
@After ---- tearDown()
@Test----testSum1()
@Test----testEven1()
```

In which order the methods will execute?

Ans - init() setup() testSum() tearDown() setUp() testEven() tearDown() close()

13. Which of the following is the correct syntax to declare the abstract method evaluate?

14. Predict the output of the below code.

```

class Car{
 void start(){
 System.out.println("Car Starts");
 }
}
class Bike{
 void start(){
 System.out.println("Bike Starts");
 }
}
class Automobile extends Car{
 void start(){
 System.out.println("Automobile Starts");
 }
}
public class ExceptionDemo {
 public static void main(String[] args){
 System.out.println("Implementing Typecasting");
 Car d = new Car();
 Automobile automobile = (Automobile) d;
 automobile.start();
 }
}

```

Ans :

Displays "Implementing type casting" and RUNTIME EXCEPTION java.lang.ClassCastException

15. Analyze the below code and predict the outcome when compiled and executed?

```

public class Demo extends Book {
 int bookid =4567;

 public int getValue() {
 return bookid;
 }

 public void call() {
 System.out.println(super.getValue()); //Line 1
 }

 public static void main(String args[]){
 Book book = new Book();
 super.call(); //Line 2
 }
}

public class Book {
 int bookId = 17897;
}

```

```

 public int getValue(){
 return bookId;
 }

}

```

Ans – Compilation error in Line2 as super keyword cannot be used in static context

16. Which of the following condition will not allow the finally block to be executed?

Ans – when System.exit(1) is called

17. What is the result of the following?

```

public class TestDemo {
 public static void main(String[] args){
 try{
 throw new ArithmeticException("AE");
 }catch(ArithmaticException e2){
 System.out.println(e2.getClass());
 }catch(Exception e1){
 System.out.println("E1");
 }
 }
}

```

Ans - class java.lang.ArithmaticException

18. What is the result of attempting to compile and run this program?

```

class CustomException extends Exception{}
class Customer extends CustomException{

public class ExceptionDemo1 {
 public static void main(String[] args){
 try{
 throw new Customer();
 }catch (CustomException customException){
 System.out.println("Custom Exception Catch block");
 }catch(Customer customer){
 System.out.println("Customer catch block");
 }
 }
}

```

Ans – Compilation error because customer class exception is not throwable

19. Which of this statement is not correct and will lead to compilation error.....

20. What will be the output of the following code when executed?

```
public class DateTimeTester {
 public static void main(String[] args){
 LocalDateTime localDateTime = LocalDateTime.of(2020,5, 13, 20, 46);
 System.out.println(localDateTime.get(ChronoField.HOUR_OF_DAY)
+localDateTime.getDayOfMonth());
 }
}
```

Ans – 33

21. Which of the below code is implemented without best practices standard?

1. String[] str=new String[]{"Hi","Hello","Welcome"};  
List strList=Arrays.asList(str);  
for(iterator itr=strList.iterator();itr.hasNext();){  
System.out.println(itr.next);
  
2. Integer i1=new Integer(11);  
Integer i2=new Integer(11);  
System.out.println(i1==i2);

Ans: Option 1 doesnot follow best practices. Can be improved using for(String s:str)

22. Which of the following is used for the automatic accurate tracking for the decimal values:

Ans:BigDecimal

23. Given:

```
public class TestDemo1 {
 public static void main(String[] args)
 {
 int i=4;
 int j=4;
 System.out.println(i==j);
 Integer w1=new Integer(4);
 Integer w2=new Integer(4);
 System.out.println(w1==w2);
 }
}
```

Ans: no issues in the above code

24. Consider the following statements:

1.

Java:

1. What will be the output of the below code?

```
//Assume all the required import statements are added
Public class TestClass
@TesT
Public void test(){
String a = "";
AssertSame("a",a);
```

**Ans: An exception is thrown at runtime as assertSame() is not a valid function**

2. What will be the output of the below code?

```
Public class Demo{
 Public static void main(String args[]){
 Int i = 34;
 Int j = 34;
 System.out.println(i==j);
 Integer i1 = new Integer(34);
 Interger i2 = new Integer(34);
 System.out.println(i1==i2);
 }
}
```

**Ans: true false**

3. Which out of the following are true in regard to interfaces in Java?

- (i) An interface can contain default and static method with defined bodies.
- (ii) An object can be created of an interface
- (iii) Multiple inheritance is allowed in Interfaces
- (iv) Multi-level inheritance is not possible in interfaces

**Ans: (i),(iii) & (iv)**

4. Given

```
Public class Sample{
 Public static void main(String[] args) throws Exception{
 Try{
 System.out.println("In try block");
 System.exit(0);
 }catch(Exception ex){
 System.out.println("In catch block");
 Ex.printStackTrace();
 }finally{
 System.out.println("In finally block");
```

```
 }
 }
}
```

Predict the output?

Ans: In try block

5. Which of the following are the advantages of exception handling in Java(Choose any 3 options)?
- (i) To maintain the normal flow of execution
  - (ii) Meaningful error reporting
  - (iii) To document compile time error
  - (iv) To prevent the abrupt termination of a program

Ans: (i),(ii) &(iv)

6. Predict the output of the below code:

```
final class Square{
 private double length,breadth;
 public Square(double length,double breadth){
 this.length = length;
 this.breadth = breadth;
 }
 Square(Square square)
 {
 System.out.println("Copy constructor invoke");
 length = square.length;
 breadth = square.breadth;
 }
 public String toString(){
 return "("+length+"+"+breadth+"i)";
 }
}
class Main{
 public static void main(String[] args){
 Square square1 = new Square(110,115);
 Square square2 = new Square(square1);
 System.out.println(square2);
 }
}
```

Ans: Copy constructor invoked  
(110.0 + 115.0i)

7. Which of the following is a valid lambda expression?

Ans: (a,b) -> {int result; return result>0;}

8. What is the output of the following code?

```
Class Employee{
 Void display(char c){
 System.out.println("Employee name starts with: " +c+ ".");
 System.out.println("His experience is : 11 years.");
 }
}

Class Main extends Employee{
 Void display(char c){
 Super.display();
 System.out.println("Another employee name also starts with: " +c+ ".");
 New Employee().display('D');
 Display(7);
 }
 String display(int c){
 System.out.println("His experience is: "+c+" years.");
 Return "Bye";
 }
}

Public class Demo{
 Public static void main(String a[]){
 Employee emp = new Main();
 Emp.display('S');
 }
}

Ans: Employee name starts with : S.
 His experience is : 11 years.
 Another employee name also starts with : S. Employee name starts with : D.
 His Experience is : 11 years.
 His Experience is : 7 years.
```

9. What is the output of the below code?

```
Import java.util.HashSet;
Public class TestDemo{
 Public static void main(String...sss){
 HashSet myMap = new HashSet();
 String s1 = new String("das");
 String s2 = new String("das");
 NameBean s3 = new NameBean("abcdef");
 NameBean s4 = new NameBean(:abcdef");
 myMap.add(s1);
 myMap.add(s2);
 myMap.add(s3);
 myMap.add(s4);
 System.out.println(myMap);
 }
}
Class NameBean{
 Private String str;
 NameBean(String str){
 This.str = str;
 }
 Public String toString(){
 Return str;
 }
}
Ans: [abcdef,das,abcdef]
```

10. Which among the following is/are true about Design Pattern

- i. Design pattern depends upon abstraction
- ii. Design patterns are completed designs that can be transformed directly into code
- iii. Design pattern depends on abstraction, follows the process of Dependency Injection
- iv. Design pattern is a template of solving problem that can be used in many real world software development

Ans i & ii

11. From the below options identify the methods and constructors in Throwable that support chained exceptions

- i. Throwable.getCause()
- ii. Throwable.initCause(Throwable)
- iii. Throwable(String, Throwable)
- iv. Throwable(Throwable)

Ans : Options i, iii & iv

12. What is Magic Number in Java in the context of java programming best practice

- i. A number which gets printed on the console
- ii. A direct usage of the number in the code
- iii. A number which magically disappears from the code
- iv. A number which is generated through error

Ans: Option2

A direct usage of the number in the code

13. Default format of LocalDate is

- I. yyyy,mm,dd
- II. yyyy.mm.dd
- III. yyyy.dd.mm
- IV. yyyy,dd,mm

Ans Option ii

14. Which among the following option are correct with respect to HashMap

- I. Override boolean equals(Object o)
- II. Override toString()
- III. Override int hashCode()
- IV. Override String hashCode()

Ans i, Override boolean equals(Object o)

15. Which of the foll interfaces are not a part of java collections framework(choose any2)

- i. List
- ii. Queue
- iii. SoretdList
- iv. ArrayList

Ans: Option iii & iv

16. Where are string objects stored in memory using “new” keyword?

- i. In Stack Memory
- ii. In String constant pool in Heap Memory
- iii. In Native Methods stack Memory
- iv. Anywhere in Heap Memory

Ans iv

17. What will be written at the line1 so that the below code will compile and run successfully?

```
Public class Main{
 //line1
 static {
 X[0] = 102;
 }
 public static void main(String [] args){
 System.out.println(x[0]);
 }
}
```

Ans : option ii, static int[]x = new int[3];

18. Given

```
public class Employee{
 private String roles;
 Public Employee (String role) {this.role= role;}
 Public Boolean equals (Employee emp){return emp.role.equals(this.role)}
}
```

Which among the following are correct with respect to the above code(Choose 2)

- I. Code give compilation error due to private attribute emp.role cannot be accessed
- II. Object.equals() method is not properly overriden
- III. Code will compile successfully
- IV. hashCode() method implementation is required

Ans iii & iv

19. Output of the below code

```
Public class TestDemo{
 Public static void main(String [] args){
 Integer n1 = new Integer (100);
 Integer n2= new Integer (100);
 Integer n3 =127;
 Integer n4 =127;
 System.out.println(n1==n2)
 System.out.println(n3==n4)
 }
}
```

Ans: false, true (option1)

20. Given

```
Public class ExceptionDemo{
 Static class Car implements AutoCloseable{
 Public void close(){
 System.out.println("Car door close")
 Throw new RuntimeException();
 }
 }

 Public static void main (String [] args){
 Try{
 //Line1

 }catch (Exception e){

 System.out.println("catch exception");
 }finally{

 System.out.println("finally")
 }
 }

 I. Car car = new Car();
 System.out.println("try block")

 II. Car car = new Car();
 Car.close();
 System.out.println("try block")

 III. Car car = new Car();

 IV. System.out.println("try block")
```

Ans: all four

21. Given an abstract class Customer as below

```
Public abstract class Customer{
 Public abstract String getCustomerType();
}
```

Select a valid implementation of getCustomerType method in another class from the below options

a. Abstract class C1 extends Customer{  
 Public abstract String getCustomerType(){ return "Premium";}  
}

- b. Customer customer = new Customer(){
 Public String getCustomerType(){ return "Premium";}
}
  
- c. class C1 extends Customer{
 Public abstract String getCustomerType(){ return "Premium";}
}
  
- d. new Customer(){
 return "Premium";
}

**Ans: a & c**

22. Given below

```

Public class Demo{
@BeforeClass
Public static void afterClass(){
System. Out.println(1)
}

@After
Public void before(){
System. Out.println(3)
}

@Test
Public void test(){
System. Out.println(5)
}

@Before
Public void fter(){
System. Out.println(4)
}
@AfterClass
Public static void afterClass(){
System. Out.println(2)
}

```

**Ans Option 1 :**

1  
4  
5  
3  
2

23. Which of the following is 'FALSE' regarding 'super' keyword in Java?

Option3, Super keyword can be used to call a parent class protected constructor which is present in the same package

24. Which of the foll data structure is used by varargs in java

Ans: Option3, ArrayList

25. Which of the following is 'FALSE' regarding 'this' keyword in Java?

Ans option4, 'this' keyword can be used to create a block of code.

26. How many number of values can be accommodated by the varargs in java

Option4, Any number of values

27. Which of the foll statement is FALSE(Choose 2 options)

- I. An interface can extend from only one interfaces
- II. A class can extend from another class and at the same time implement any number of interfaces
- III. A class can extend multiple abstract classes
- IV. Many classes can implement the same interface

Ans i and iii

28. Have a look at the foll code and choose the correct option

```
Class ReportUtil{
 Int reportId =0;
 Static {++reportId}
 Static int employeeReportId;
 Public void preReport(){
 employeeReportId= reportId;
 }
}
```

Ans: option 3 The code will not get compiled as instance variable cannot be referred from the static block

29. Which of the below methods can be defined in the single class in java. Select most suitable(select 2 options)

- 1. Void add(int,int)
- 2. Void add(int,int, int)
- 3. int add(int,int)
  - I. 1 & 3 together
  - II. 1 & 2 together
  - III. 2 &3 together
  - IV. 1,2 &3 together

**Ans Options ii and iii**

30. Ria has a class called Account.java under two separate packages com.infy.debit and com.infy.credit. Can she use the Account class of both the packages in another class called ReportUtil.java of package com.infy.util

**Option2: No, She cannot as there will be a compilation error stating that the import collides with another import**

31. Which of the following are true about enums in Java? (Choose 3 correct)

- I. enums can implement any other interface in java
- II. an instance of enum can be created outside of enum itself
- III. enum cant extend any other class except the abstract base class java.lang.enum
- IV. enum can be used to implement singleton design pattern

**Ans i, iii & iv**

32. What's is true with respect to abstract class being given below

```
Abstract class Employee{
 //fields and constructor
 Public void salaryComputer()
 {
 // code goes here
 }
 Public abstract void taxReduce()
 {
 // code goes here
 }
 Public abstract void benifitsInclude();
}
```

**Ans: option4. Abstract method of class Employee has definition**

33. Which of the following are NOT good practice for creating objects.

- a. Lazy initialization of objects
- b. Creating String literals instead of String objects
- c. **Creating Wrapper objects instead of primitives**
- d. invoking static factory methods for immutable classes

**Ans option c**

34. Which among the foll are valid lamda expressions to sort the numbers in numberlist in descending order choose 3

- a. numberList.sort((x,y)->x.compareTo(y))
- b. numberList.sort(int x,int y)->x.compareTo(y))
- c. numberList.sort((x,y)->{return x.compareTo(y)})
- d. numberList.sort((Integer x, Integer y)->x.compareTo(y))

Ans

35. What is the output for the below code

```
public class VarArgsDemo{
 static void func(int...x)
 {
 System.out.println("Number of arguments "+x.length);

 for(int i:x)
 System.out.print(i+"");
 System.out.println();
 }
 void func(int a) //Line1
 { System.out.println("one"); }

 public static void main(String[]args){
 new VarArgsDemo().func(150);
 func(11, 12, 13, 14);
 func(); }
 }
Ans:Option B
```

```
Number of arguments1
150
Number of arguments4
11121314
Number of arguments0
```

36.Given, Assume all the required imports are added

```
Public class TestDemo{
 Private ArrayList list //Line1
 @Test //lin2
 Public void test(){
 assertTrue(list.isEmpty());
 }
}
```

Which of the following is true regarding the above code fragment

```
Ans:if we replace Line2 by “@Test(expected = NullPointerException.class)” the test
case will pass
```

37. Given

```
HashMap <Integer, Integer> myMap = new HashMap<Integer, Integer>();
myMap.put(1001,5);
myMap.put(1002,8);
myMap.put(1002,5);
myMap.replace(1002,5, 100);
system.out.println(myMap)
```

```
Ans: {1001=5, 1002=100}
```

38. What is the result when the foll code is compiled?

```
Class Student extends Exception{}
Class Hosteller extends Student{}
Public class StudentTester{
 Public static void main (String args[]){
 Try{
 //some monitored code
 Throw new Hosteller();

 }Catch(Student st){
 System .out.println("Student class Exception");
 }catch (Hosteller host){
 System .out.println("Hosteller class Exception");
 }
 }
}
```

**Ans: The code will not compile Unreachable catch for Hosteller because Student class exception is caught before Hosteller**

39.

```
class Operations{
 Public void addition(){
}
```

```
Class AdvOperations extends Operations{
Void addition()//line1
```

```
{
}
```

Line 1 generates compilation error . which of then below option helps to resolve this

**Ans: public keyword has to be added**

40. What will be in line1 to get the output as 150. Choose 2

```
class EmployeeUtil {
 //|line1
 double amountTax;
 public double taxCalculator(double salary)
 {
 amountTax = salary *TAX_PERCENTAGE/100;
 return amountTax;
 }
}
public class ConstantMain
{
```

```

public static void main(String[] args) {
 System.out.println(new EmployeeUtil().taxCalculator(5000));
}
}

```

Ans:

```

final int TAX_PERCENTAGE=3;
static int TAX_PERCENTAGE=3;

```

41.

```

interface Component {
 String cname = "Motor";
 String getName(String name);
}

public class Demo implements Component{
 public String getName(String name) {
 System.out.println("Inside Demo Class");
 return "Componenet from interface is :" + cname+"and component
from class is "+ name+ ";";
 }

 public static void main(String[] args) {

 Demo demo = new Demo();
 System.out.println(demo.getName("Battery"));
 demo.getName("Battery");

 }
}

```

Ans Inside Demo Class  
 Componenet from interface is :Motorand component from class is Battery;  
 Inside Demo Class

42.

```

public class CollectionTest {

 public static void main(String[] args) {

 Collection<Integer> employeeCollection = new HashSet();
 collection.add(1001);
 collection.add(1002);
 collection.add(1001);

 Set<Integer> newSet = new TreeSet();
 }
}

```

```

 newSet.addAll(employeeCollection);
 System.out.println(newSet);

 }
}


```

**Ans : Runtime Exception : null pointer Exception**

43. Which of the following regarding an abstract class is/are true in Java

- i) Object of an abstract class cant be created
- ii) An abstract class is designedonly to act as a base class in hierarchy to be inherited by other classes

**Ans : Both i & ii**

44.

```

public class CodeForException {
 //public static void main(String[] args) {
 public void callMe() throws Exception{
 try
 {
 int value=3/0;
 }
 catch (ArithmeticeException aa)
 {

 System.out.println(aa);
 }
 }
 public void calling()
 {

 callMe();
 }
}

```

**Ans : Compilation error**

45.

```

public class Demo {
 private static String id;
 private Random random = new Random();

 public Demo(){

 if(id ==null){
 }
 }
}

```

```

 id = "ACC1101"+ Math.abs(random.nextInt());
 }
}

public String getid() {
 return id;
}

}
Ans - if(id==null){

```

**46. Given**

```

//Assume all the required imports are added

public class TestDemo {
 private ArrayList list;//Line1
 @Test //Line2 (expected = NullPointerException.class)
 public void test(){
 assertTrue(list.isEmpty());
 }
}

```

Which of the following statement is true regarding the above code fragment?

Ans: if we replace Line2 by “@Test(expected = NullPointerException.class)” the test case will pass

**47.**

```

public static void main(String args[])
{
 HashMap<Integer, Integer> myMap = new HashMap<Integer, Integer>();
 myMap.put(1001, 5);
 myMap.put(1002, 8);
 myMap.put(1002, 5);
 myMap.replace(1002, 5, 100);
 System.out.println(myMap);
}

```

Ans: {1001=5, 1002=100}

48. what is the result when the following code snippet is compiled?

```

class Student extends Exception{}
class Hosteller extends Student{}
public class StudentTester {
 public static void main(String[] args){
 try{
 //some monitored code
 throw new Hosteller();
 }
 catch (Student st){

```

```

 System.out.println("Student class exception");
 }
 catch (Hosteller host){
 System.out.println("Hosteller class exception");
 }
}
}

```

**Ans:** The code will not compile Unreachable catch for Hosteller because Student class exception is caught before Hosteller

49.

```

class Operations
{
 public void addition()
 {}
}
class AdvOperations extends Operations {
 void addition()//Line1
 {}
}

```

Line 1 generates Compilation error. Which of the below options helps to resolve this?

**Ans:** public keyword has to be added

50.

```

Public class TestDemo{

Public void main(int x){

System.out.println("Main1")

}

Public static void main(String args[]){

System.out.println("Hello Main")

}

}

```

**Ans :** option 4 : Hello Main

51.

```
import java.util.HashMap;
import java.util.Iterator;
import java.util.TreeSet;
public class ABC {
 public static void main(String args[])
 {
 TreeSet set = new TreeSet();
 set.add("a");
 set.add(6);
 set.add("c");
 Iterator ite =(Iterator) set.iterator();
 while(ite.hasNext()){
 System.out.println(ite.next()+"");
 }
 }
}
```

Ans: The code will give `java.lang.ClassCastException: java.lang.String incompatible with java.lang.Integer`

52. what is the output of the following code?

```
public class Demo5 {
 public static void main(String[] args){

 int x[][] = new int [4][]; //Line 1
 x[0] = new int [1];
 x[1] = new int [2];
 x[2] = new int [3];
 x[3] = new int [4];
 int a,b,c=0;
 for(a=0; a<4; a++){
 for(b=0; b<a+1; b++){
 x[a][b] = c;
 c++;
 }
 }
 for(a=0; a<4; a++){
 for(b=0; b<a+1; b++){
 System.out.print(" " + x[a][b]);
 c++;
 }
 }
 System.out.println();
 }
}
```

```
}
```

Ans:

```
0
12
345
6789
```

53. which of the following keyword is used to prevent the content of a variable from being modified from outside.

Ans - Final

54.

```
class Book{
 int bookid =2356;

}
class Book1 extends Book{
 int bookid = 1167;
}
class Book2 extends Book1{
 int bookid = 2378;//Line8
 void display(){
 System.out.println(super.super.bookid);//Line10
 System.out.println(super.bookid);//Line11
 System.out.println(bookid);
 }
}
public class Demo2 {
 public static void main(String[] args){
 Book2 book2 = new Book2();
 book2.display();
 }
}
```

Ans - Compilation fails, because of an error in Line10 as super keyword is unexpected.

55. Consider the below class and identify the extension of the output file when we execute the command javac Employee.java

```
class Employee1 {
 private int x=10;
 public void showX(){
 System.out.println(x);
 }
}
```

Ans - .java

56. which of the below if statement is used to find a year is a leap year or not

```
Ans: if((y%4==0) && (y%100 !=0) || (y%400==0))
```

57. Analyze the below code and predict the output when executed the code

```
public class Demo6 extends Book{
 int bookid = 4362;
 public int getvarchar(){
 return bookid;
 }
 public void Cell(){
 System.out.println(getvarchar()); //Line1
 }
 public static void main(String[] args){
 Book book = new Book();
 super.Cell(); //Line 2
 }
}
class Book{
 int bookid = 17252;
 public int getvarchar(){
 return bookid;
 }
}
```

```
Ans: Compilation error in line 2 as super keyword cannot be used in static context
```

58. have a look at the following class and predict what should be placed at line 1 to get 150.0 as output when the code gets executed? (choose 2 options)

```
class EmployeeUtil{
 //line1
 double amountTax;
 public double taxCalulate(double salary){
 amountTax = salary*TAX_PERCENTAGE/100;
 return amountTax;
 }
}
public class ConstantMain{
 public static void main(String[] args){
 System.out.println(new EmployeeUtil().taxCalulate(5000));
 }
}
```

Ans:

```
static int TAX_PERCENTAGE=3;
final int TAX_PERCENTAGE=3;
```

**59.** if the child class of an abstract class does not implement all its abstract methods then it should be declared as

**Ans: Option A: Abstract class**

60. Which of the following statements regarding an abstract class is/are true in Java

- I. Object of an abstract class cant be created
- II. An abstract class is designed only to act as a base class in hierarchy to be inherited by other classes

**Ans : Both I & II**

61. which of the following is valid function used to read values using Scanner in java (choose 3)

1. **nextInt()**
2. **nextChar()**
3. **nextLong()**
4. **nextLine()**

**Ans: option 1,3 and 4**

62. What will be output of the below code snippet?

```
package Example;

public class Pet {
 public void displayName(){
 System.out.println("Inside Pet");
 }
}

package java.pac;
import Example.Pet;
public class Dog extends Pet{
 @Override
 public void displayName(){
 System.out.println("Inside Dog");
 }
}
```

```

 }
}

package com.infy;
import Example.Pet;
public class Demo7 {
 public static void main(String[] args){
 Pet pet = new Dog(); //Line1
 pet.displayName();
 }
}

```

}

**Ans – Compilation error at Line1 due to ClassCastException**

63. Which of the foll is FALSE regarding polymorphism in JAVA

- a. Polymorphism is the ability of an object to take different forms
- b. Polymorphism can be as a single action that can be performed in different ways
- c. Polymorphism means different forms by creating different classes that are referred to each other by inheritance
- d. Polymorphism is using same method multiple times in a class with different parameters

**Ans Option D**

64. what is the result of the following code?

```

public class Vehicle {
 static class Car{
 public void go(){
 System.out.println("Car Ignition");
 }
 }
 static class ElectricCar extends Car{
 public void go(){
 System.out.println("ElectricCar Ignition");
 }
 }
 static class PetrolCar extends Car{
 public void go(){
 System.out.println("PetrolCar Ignition");
 }
 }
 public static void main(String[] args){
 Car car = new ElectricCar();
 car.go();
 }
}

```

**Ans- ElectricCar Ignition**

65. If child class of abstract class doesnot implement all its abstract methods then it should be declared as?

**Ans - abstract class**

66. which are valid upper bound by class Employee of list. (Doubt)

**Ans - List<?extends Employee>**

**Choose any 2**

```
public class App {
 public static void main(String[] args) {
 String msg = null;
 try {
 System.out.println(msg.length());

 }catch(NullPointerException ex) {
 System.out.println("Exception is caught here");

 throw ex; //Line 1
 System.out.println(msg); //Line 2
 }
 }
}
```

**Ans:**

**//a, b**

**A.Place line2 before line 1**

**B.Remove line 2 ie after throw, there should not be any statements**

```
interface Book {
 static void bookName() {
 System.out.println("in interface book");
 }
}

public class BookImpl implements Book{
 void bookName() {
 System.out.println("In BookImpl Class");
 }
}
```

```
}
```

```
public class BookApp {

 public static void main(String[] args) {
 // TODO Auto-generated method stub
 new BookImpl().bookName();
 }

}
```

ANS: code compiles and prints “In BookImpl Class”

Which of the below is not a valid classification of design pattern

Creational Pattern

Structural pattern

Behavioral Pattern

Java pattern

ANS: D Java pattern

### Section 1:

1. What is the output of the following code?

```

public class Main{
 public static void main(String args[]){
 int twoD[][] = new int [4][];
 twoD[0] = new int [1];
 twoD[1] = new int [2];
 twoD[2] = new int [3];
 twoD[3] = new int [4];
 for (int i=0;i<4;i++){
 for (int j=0;j<i+1;j++){
 twoD[i][j]; //Line 2
 }
 System.out.println("executed");
 }
}
}

```

**Ans – Compilation error in Line 2 as there is no Assignment operator used**

2. Which of the following statement is false about an object in java?

**Ans – Object can communicate with each other**

3. What is the output of the code given below?

```

public class ABC {
 public static void main(String args[])
 {
 boolean flag = false;
 if(flag = true)
 {
 System.out.println("true");
 }
 else
 {
 System.out.println("false");
 }
 }
}

```

**Ans- true**

4. Ria has a class called ‘Account.java’ under two separate packages. Com.infy.debit and com.infy.credit can see give the account.java of both the packages in another class called ‘ReportUtil.java’ of package com.infy.util?

**Ans – No She cannot as there will be compilation error stating the import collides with another import.**

5. Identify the suitable datatype to be used for variable ‘b’ at line 1?

```

class Expression {
 public void calc()
 {

```

```

 double x=10;
 int y = 20;
 float z= 30;
 //Line 1 --- Answer is double
 b=x+y+z;

 }

}

```

**Ans – double**

6. What is the result when the following code is compiled and executed?

```

class Demo {
 int x=1;
 int y=2;
 Demo display(Demo demoParam){
 Demo obj = new Demo();
 obj = demoParam;
 obj.x = demoParam.x++ + demoParam.y ;
 demoParam.y = demoParam.y;
 return obj;
 }
 public static void main(String[] args){
 Demo obj1 = new Demo();
 Demo obj2 = obj1.display(obj1);
 System.out.println("obj1.x = " + obj1.x + "obj1.y = " + obj1.y);
 System.out.println("obj2.x = "+ obj2.x + "obj1.y = " + obj2.y);

 }
}

```

**Ans:**

obj1.x = 4obj1.y = 3  
 obj2.x = 4obj1.y = 3

7. Have a look at the following class and predict the option that is correct?

```
public class CodeForException {
```

```

public void callMe() throws Exception{
 try{
 int value = 3/0;

 }catch (ArithmaticException ae){
 System.out.println(ae);
 }
 }
public void calling(){
 callMe();
}
}

```

**Ans:** The code will face issues during compilation as the calling code neither handles nor throws Exception

8. What will be output of the code when executed?

```

public class OperatorTester {
 public static void main(String[] args){
 int a = 10;
 int b = 15;
 if (++a <(b=b-=4) || (a=a+=4)>b++){
 System.out.println(a+", "+b);
 }
 }
}

```

**Ans – 15,12**

9. What is the result when the following code snippet is executed?

```

class Employee {
 static String name = " ";
 public static void main(String[] args) throws Exception{
 try{
 name+="jo";
 throw new Exception();
 }

 }catch(Exception e){
 name+="hn";

 }finally{
 name+="s";
 empName();
 name+="on";
 }
 }
}

```

```

 }
 System.out.println(name);
 }
 static void empName(){
 throw new NullPointerException();
 }
}

```

**Ans –** The code will give java.lang.NullPointerException in finally block when executing

- What is the result when the code is compiled and executed?

```

class Calculator {
 int a = 123, b=200;
 public void display(){
 System.out.println("a:" + a + "b:" + b+"");
 }
}
class CalculatorDemo{
 public static void main(String[] args)
 {
 Calculator calculator1 = new Calculator(); //Line1
 Calculator calculator2 = calculator1; //Line2
 calculator1.a+= 1;
 calculator1.b+= 1;
 System.out.println("calculator1 values:");
 calculator1.display();
 System.out.println("calculator2 values:");
 calculator1.display();
 }
}

```

**Ans –**

```

calculator1 values:
a:124b:201
calculator2 values:
a:124b:201

```

## Section2

- What can be expected when the following code is compiled

```

abstract class Customer {

 public int custId;//Line1
 Customer()//Line2
 {
 custId=23456;//Line3
 }
 abstract public void setId();
 abstract final public void getid();//Line4
}

```

**Ans : Option D: Compilation error in Line 4 abstract methods cant be final**

2. What is the correct way of placing ‘this’ keyword in a constructor?

**Ans Option A : First Statement**

3. At what position should Varargs be placed in a parameterised method

**Ans Option B: Last Place**

4. Predict the output of the following code

```

public class Demo {
 static int x=232;
 int y=135;
 public void display(){
 System.out.print("Inside Demo");
 }
 public static void show(){
 System.out.print(x);
 }

 public static void main(String[]args)
 {
 Demo.show(); //Line 1
 Demo demo=new Demo();
 demo.show(); //Line2
 show(); //Line3
 demo.display();
 }
}

```

**Ans: Option A: Compilation error in line2 as static method cannot be called with object reference**

5. Which of the following keyword can be used to restrict a class to be implemented in java

**Ans Option B. Final**

6. Given below. What will be the output when code is compiled and executed

```

public class Parent {

 public final void show(){
 System.out.println("show() inside Employee");
 }
}

final class Child extends Parent {
 public void show1() { //Line1
 final int x=100;
 System.out.println("show() inside Unit");
 System.out.println(x);
 }
}

public class Demo {

 public static void main(String[] args) {
 Parent parent = new Child();
 new Child().show1(); //Line2
 }
}

```

Ans: Option D

**show() inside Unit**

**100**

7. Which of the following is FALSE regarding parameterized constructor in Java

- a. parameterised constructor should have void as return type
- b. parameterised constructor can take any number of parameters
- c. parameterised constructor cannot have private access modifier
- d. parameterised constructor throw an exception

Ans Option c parameterised constructor cannot have private access modifier

8. What is the output of the following

```

public class Trainer{

 public void display(String name) {
 System.out.println("I am a trainer");
 print(name);
 }

 public void print(String name) {
 System.out.println(" I train "+ name+ ".");
 }
}

```

```

public class Trainee extends Trainer {
 String myname;

 public Trainee(String myname) {
 super();
 this.myname= myname;
 }

 public void display(String name) {
 super.display(name);
 System.out.println("I am a trainee");
 print("Java");
 }

 public void print(String name) {
 super.print(name);
 System.out.println("I want to learn " +name+"");
 }

 public static void main(String[] args) {
 Trainer trainee = new Trainee("XYZ");
 trainee.display("Java");

 }
}

```

Ans: Option B

```

I am a trainer
I train Java.
I want to learn Java
I am a trainee
I train Java.
I want to learn Java

```

9. Which of the following is FALSE regarding ‘this’ keyword in java?

Ans--- optionD, ‘this’ keyword can be used to create a block of code.

10. Given

```

Class Parent{}
Class Child extends Parent{}
Final class GrandChild extends Child{}

```

Which of the following statement is not 'true' about the above code.

Ans : Option C: Reference of parent class can accept the instance of child class but not the instance of GrandChild class

## Section 3

1. What will be the output of the below code

```
public class student {

 String stuName="Jacklin";

 void func() throws Exception{
 try
 {
 stuName+="--";

 }
 catch (Exception e) {
 throw new Exception();
 }
 finally
 {
 stuName+="Hello" +stuName;
 }
 stuName+="!!!!";
 }

 void disp() throws Exception
{
 func();
 System.out.println(stuName);
 }

 public static void main(String[] args) {
 try
 {
 student student = new student();
 student.disp();
 }
 catch (Exception e) {
 System.out.println("CatchBlock");
 }
 }
}
```

Ans Jacklin--HelloJacklin--!!!!  
Option D

2. Predict the output of the foll

```
public class Greeter{

 public static void main(String[] args) throws Exception{
 try
```

```

 {
 System.out.println("Greeting"+ ""+args[0]);

 }catch (ArrayIndexOutOfBoundsException e) {

 System.out.println("Sam");
 }

}

```

Ans : Option C Sam

3. Predict the output of the below code.

```

public class TestDemo{

 public static void main(String[] args) {
 for(int a=0;a<6;++)
 try {
 if(a%3==0)
 throw new Exception("Except1");
 try {
 if(a%3==1)
 throw new Exception("Except2");
 System.out.println(a);

 }catch (Exception inside) {
 a +=2;
 }finally {
 ++a;
 }
 }catch (Exception outside) {
 a+=3;
 }finally {
 ++a;
 }
 }
 }
}

```

Ans: Option A: 5

4. Given

```
Public class Sample{
 Public static void main(String[] args) throws Exception{
 Try{
 System.out.println("In try block");
 System.exit(0);
 }catch(Exception ex){
 System.out.println("In catch block");
 Ex.printStackTrace();
 }finally{
 System.out.println("In finally block");
 }
 }
}
```

Predict the output?

Ans: Option B.In try block

5. Which of the following exceptions are ignored during compile time (choose any 2 option)

Option A and C

ArrayIndexOutOfBoundsException

NullPointerException

## Section 4

1. Identify the outcome of the given code snippet

```
public class Demo{

 public static void main(String[] args) {
 int [] arrVar = {11,22,33,44,55,66,77,88,99,109};
 int position =3;
 int value =7;
 for(int i= arrVar.length-1;i>position;i--) {
 arrVar[i]= arrVar[i-1];
 }
 arrVar[position]= value;
 System.out.println("New Array"+ Arrays.toString(arrVar));
 }
}
```

Ans :

Option D New Array[11, 22, 33, 7, 44, 55, 66, 77, 88, 99]

2.Which of the following are true about enums in Java? (Choose 3 correct)

- I.     enums can implement any other interface in java
- II.    an instance of enum can be created outside of enum itself

- III. enum cant extend any other class except the abstract base class java.lang.enum
- IV. enum can be used to implement singleton design pattern

Ans Options i, iii & iv

3. Predict the output of the below

```
public static void main(String[] args) {
 String name1 = "Java";
 String name2 = "Java";
 System.out.println(name1==name2);
 System.out.println(name1.equals(name2));
}
```

Ans Option A: True True

4. Given

```
public class App{

 public static void main(String[] args) {
 String s1 = new String ("smart");
 String s2 =s1;
 if(s1==s2) {
 System.out.println("==smart");
 }
 if (s1.equals(s2)) {
 System.out.println("equals smart");
 }
 }
}
```

Ans Option B

==smart  
equals smart

## Section 5

1.What will happen when the following code is executed? ( Output - not sure)

```
public class TestDemo{

 public static void main(String[] args){
 List list1=new ArrayList<>();
 list1.add("1");
 list1.add("2");
 list1.add(1, "3"); //Line1
 List list2=new LinkedList<>(list1); //Line2
 list1.addAll(list2); //Line3
 list2=list1.subList(2,6); //Line4
 list2.clear();
 System.out.print(list1+ "");
 }
}
```

Ans: [1, 3] (answer not in option)

2. Which among the following option are correct with respect to HashMap

- Override boolean equals(Object o)
- Override toString()
- Override int hashCode()
- Override String hashCode()

Ans – option a - Override boolean equals(Object o)

3. Identify the incorrect statement as per the collection Framework hierarchy?

Ans D: LinkedHashSet Implements HashSet

4. What is the result of compiling and running this code snippet?

```
import java.util.Arrays;
import java.util.Comparator;

public class Demo3 implements Comparator<String>{
 @Override
 public int compare(String x, String y){
 return x.toLowerCase().compareTo(x.toLowerCase());
 }
 public static void main(String[] args){
 String[] values = {"JOHN", "Annie", "JACKLINE"};
 Arrays.sort(values,new Demo3());
 for (String str:values)
 System.out.print(str + " ");
 }
}
```

Ans - JOHN Annie JACKLINE

5. Predict the output of the below code snippet?

```
import java.util.ArrayList;
import java.util.List;

public class sam1 {
 public static void main (String[] args){
 List<String> list = new ArrayList();
 list.add(0,"A");
 list.add(1,"B");
 list.add(1,"C");
 for (Object object:list){
 System.out.print(" " + object);
 }
 }
}
```

Ans --- A C B

## Section 6:

1.What is true regarding the following code snippet

```
public interface StaticInterface {

 static void staticMethod()
 {
 System.out.println("Inside interface");
 }
}

public class StaticInterfaceImpl implements StaticInterface {

 public void staticMethod()
 {
 System.out.println("Inside class");
 }
}

public class StaticDemo {

 public static void main(String[] args)
 {
 new StaticInterfaceImpl().staticMethod();
 }
}
```

```
}
```

Ans: Option D. code will print “inside class” on execution

2. Which out of the following are true in regard to interfaces in Java
- a. An interface can contain default and static method with defined bodies
  - b. An object can be created of an interface
  - c. Multiple inheritance is allowed in interface
  - d. Multi level inheritance is not possible in interface

Ans Option A and C

3. Default format of LocalDate is

Ans Option B yyyy.mm.dd

4. Given

```
interface Greeting{
 default void greet() {
 System.out.println("In Greet interface");
 }
}

class GreetingDef implements Greeting{
 public void greet() {
 System.out.println("In GreetingDef class");
 }
}

public class App {

 public static void main(String str []){
 Greeting obj = new GreetingDef();
 obj.greet();
 }
}
```

Ans:Option B- In GreetingDef class

5. The below code will generate compilation error. Select the possible options to avoid it(Choose 2)

```
public interface Insurance{
```

```

static void policy() {
 System.out.println("policy");
}

public class InsuranceImpl implements Insurance{
 public void policyPayment() {
 policy();
 }
}

public class App{
 public static void main(String[] args) {
 new InsuranceImpl().policyPayment();
 }
}

```

- a. Static method of an interface can only be accessed using interfaces name
- b. Static method should always be public
- c. Static method cannot be invoked inside the non static method
- d. Policy() method of interface has to be accessed using interface name

**Ans option A & D**

## **Section 7:**

1. What is Magic Number in java in the context of java programming best practices?

**Ans – A direct usage of number in the code.**

2. Given

```

public class Employee {
 private int empId;
 private String empName;
 private String designation;
 private transient String dob;
}

```

Analyze the given code and identify the suitable comments from the below option.

Ans:

- (i) Fields in non-serializable classes should not be 'transient'
- (ii) Make the Employee class as serializable

3. From the below options identify the methods and constructors in Throwable that support checked exception?

- (i) Throwable getCause()
- (ii) Throwable initCause(Throwable)
- (iii) Throwable (String, Throwable)
- (iv) Throwable (Throwable)

Ans - (i), (iii), (iv)

4. Identify the valid code needs to be inserted in line5, assume the code is running in

```
import java.util.Random;
```

```

public class Emp {
 private static String id;
 private Random random = new Random();
 public Emp(){
 //line5 if(id==null){
 id = "ACC1101" + Math.abs(random.nextInt());
 }
 }
 public String getId(){
 return id;
 }
}

```

Ans - if(id==null){

5. Select the valid code fragment according to Java coding standard?

- (i) **public void draw(String s){**  
    **if(s.equals("Square")){**

```

 drawSquare();
 }
}
(ii) public void draw(String s){
 if("Square".equals(s)){
 drawSquare();
 }
}

```

**Ans:** Both (i) and (ii) are valid.

## Section 8:

1. Which among the following comes under Creational Design pattern?

**Ans:** Singleton Design Pattern

2. Which of the statements are true about design patterns?
  - i. There are only three design patterns defined for Java
  - ii. We can use each design pattern only once per application
  - iii. Design patterns are conceptual reusable solutions

**Ans:** Statement i,ii,iii

3. What changes need to be made in the following code to make the singleton pattern correct?  
(choose 2)

```

public class Employee1 {
 public static Employee1 employeeInstance;
 private Employee1(){}
 public static Employee1 getEmployee()
 {
 if (employeeInstance==null){
 employeeInstance = new Employee1();
 }
 return employeeInstance;
 }
}

```

**Ans:**

Option A &D

- A. None the singleton pattern is properly implemented
- D. Change the access modifier of employeeinstance from public to private

## Section 9:

1. Given

```

//Assume all the required imports are added
import java.util.ArrayList;

import org.junit.Before;
import org.junit.Test;

public class Demo4 {
 static int a = 10;
 static ArrayList b = new ArrayList();
 @Before
 public void init(){
 a=15;
 b.add(a);
 }
 @Test
 public void test(){
 a=a+20;
 System.out.print(a);
 System.out.println(b);
 }
 @Test
 public void test1(){
 a=a+30;
 System.out.print(a);
 System.out.println(b);
 }
}

```

Predict the output?

Ans -  
 35[15]  
 45[15, 15]

2. Which of the following annotation must be used in a test class to run same test again and again

Ans: @Test

3. Predict the output for the below

```

//Assume all the required import statements are added
import org.junit.Before;
import org.junit.Test;

import junit.framework.Assert;

public class TestDemo1 {
 @Before
 public void beforeTest1(){
 System.out.println("in before test2");
 }
 @Before

```

```

public void beforeTest2(){
 System.out.println("in before test1");
}
@Test
public void test(){
 String a = "123";
 Assert.assertEquals("123",a);
}

```

**Ans:**

Test Passed as they point the same object and prints the below in console  
 in before test1  
 in before test2

ThJava Question

- Which of the following is not pre defined annotation in Java?

- @Deprecated
- @Overriden
- @SafeVarargs
- @FunctionInterface

**Ans: Option B(@Overriden)**

- public class TestString3{  
 Public static void main(String[] args){  
 //insert code here//Line3  
 System.out.println(s)  
 }  
 }

Which of the below code fragment when inserted independently at line3 generate output as 498

**Ans Option B**

```
StringBuffer s = new StringBuffer("123456789").s.delete(0,3).replace(1,3,"98").delete(3,8);
```

3.Predict output of this

```

public static void main(String[] args) {

 String Value1= "Hello";

```

```

String Value2= new String ("Hello");
System.out.println(Value1.equals(Value2)+" "+(Value1==Value2));
String Value3= Value2.intern();
System.out.println((Value1==Value3)+" "+(Value1.equals(Value3)));
}

}

```

**Ans truefalse**  
**Truetrue**

66. Which of the following statements are true if a duplicate element obj T is added to a HashSet?

- a) The element obj T is not added and add() method returns false
- b) The element obj T is added successfully
- c) An exception occurs during runtime
- d) An exception occurs during compile time

. which of the following is incorrect regarding interfaces in Java SE8

- a.all the methods are public,abstract by default
- b.all the variables are public by default
- c.methods can have implementation
- d.its possible to hold static method

- a) a and b
- b) b and c**
- c) a,b and c
- d) a only

Which of the below are NOT good practices for creating objects?

- a) Lazy initialization of objects
- b) Creating String literals instead of String objects
- c) Creating Wrapper objects instead of primitives**
- d) invoking static factory methods for immutable classes

Which of the below statement indicate the need to use the factory pattern?

- a) we have two classes that do the same thing
- b) we only want one instance of the object to exist
- c) we want to build a chain of objects
- d) we don't want the caller to depend on a specific implementations**

4.What is the output when the below code is compiled and executed

```
Package exceptions;
Public class Demo
{
public static void main(String[] args) {

 try
 {
 return;

 }
 finally
 {
 System.out.println("Finally");
 }

}
```

**Ans : Option A:( Finally)**

What is the output of the below code snippet

```
enum Customer
{
private CUSTID
public CUSTNAME
protected ADDRESS

}
```

**Ans : D Compilation Fails**

(Explanation: Enum cannot have any modifiers. They are public, static and final by default)

What is the output of the below code

```
package javaBasics;
public class student {

 String stuName="Jackin";
 void display()
 {
 try
 {
 stuName+="John";
 func();
 }
 catch (Exception e) {
```

```

 stuName+="GoodName";
 }
 }

void func() throws Exception{
 try
 {
 stuName+=".....";
 method();
 }
 catch (Exception e) {
 throw new Exception();
 }
 finally
 {
 stuName+="!!!!!";
 }
 stuName+="hello";
}
void method() throws Exception
{
 throw new Exception();
}

void disp()
{
 System.out.println(stuName);
}

public static void main(String[] args) {
 try
 {
 student student = new student();
 student.display();
 student.disp();
 }
 catch (Exception e) {
 System.out.println("CatchBlock");
 }
}
} Ans Option D (JackinJohn.....!!!!GoodName)

```

What will be the output of the following code

```

public class Test{
 public void method()
 {
 for(int i=0;i<3;i++) {
 System.out.println(i);
 }
 System.out.println(i);
 }
}

```

```
 }
}
```

**Ans: C. Compilation Fails**

What is the result of attempting to compile and run this program

```
public class Customer{
 public static void main(String [] args)
 {
 Float f1= new Float(67.65f);
 Float f2= new Float(36.45f);
 if(f1>f2)
 {
 System.out.println("f1 is bigger than f2");
 }
 else
 {
 System.out.println("f1 is not bigger than f2");
 }
 }
}
```

**Ans : Option A (f1 is bigger than f2)**

What is the result of the following code snippet when compiled

```
public class Employee {
 int employeeid;
 double getEmployeeid()
 {
 System.out.println("Employee Id");
 return employeeid;
 }
}
```

Ans A: The code will not be complied as there is no main method

Given Enum definition and java class

```
enum Day{
```

```

 SUNDAY(1), MONDAY(2), TUESDAY(3), WEDNESDAY(4), THURSDAY(5),
FRIDAY(6) ,SATURDAY(7) ;
private int value;
private Day(int value)
{
 this.value =value;
}

public int getValue()
{
 return this.value;
}

}

public class Employee {
 public static void main(String[] args)
 {
 for (Day day: Day.values()) {

 //Line1
 System.out.print(day.toString()+"-");

 //System.out.print(day.name()+"-");

 }
 }
}

```

What should be [placed in line 1 to get the output as  
SUNDAY-MONDAY-TUESDAY-WEDNESDAY-THURSDAY-FRIDAY-SATURDAY-  
Choose one or more options

**Ans : A and C**

```

System.out.print(day.toString()+"-");
System.out.print(day.name()+"-");

```

What will be the output of the following code

```

public void method(){
 for(int i=0;i<3;i++){
 System.out.print(i);
 }
 System.out.print(i);
}

```

**Ans: C. Compilation fails**

which of the below exceptions are mostly thrown by JVM in a Java application?(Choose all that apply)  
means runtime exception

- a) ClassCastException
- b) IllegalStateException
- c) NumberFormatException
- d) IllegalArgumentException
- e) ExceptionInitializerError

Checked Exceptions are Compile time exceptions

Unchecked Exceptions are runtime exceptions

//Check Tutorial

What's the output of the following code

```
public class Demo {
 void main(){
 System.out.println("JAVA");
 }
 static void main(String args){
 System.out.println("Spring");
 }
 public static void main(String[] args){
 System.out.println("Hibernate");
 }
 void main(Object[] args){
 System.out.println("Apache Camel");
 }
}
```

**Ans Option A: Hibernate**

Given the below code snippet, predict the correct option

```
public class Operator {
 public static void main(String[] args){
 float val1=5.3f;
 float val2=2.3f;
 float result= val1 %val2;
 System.out.println(result);
```

```
 }
}
```

Ans: Option A **Code compiles, runs and produces the output 0.7000003**

What is the output for the below code

```
public class Employee {

 public final void show(){
 System.out.println("show() inside Employee");
 }

}

public class Dem011 {

 public static void main(String[] args) {
 Employee employee = new unit();
 new unit().show1();

 }
}

final class unit extends Employee {
 public void show1() {
 final int x=100;
 System.out.println("show() inside Unit");
 System.out.println(x);
 }
}
```

Ans:Option D.  
**show() inside Unit  
100**

What is the output when we execute the below code

```
public class Dem011 {
 static class Customer {
```

```

 public void go() {
 System.out.println("Inside Customer");
 }
 }
 static class Account extends Customer {
 public void go() {
 System.out.println("Inside Account");
 }
 }
 static class Branch extends Customer {
 @Override public void go() {
 System.out.println("Inside Branch");
 }
 }
 public static void main(String[] args) {
 Customer customer = new Account();
 Branch branch = (Branch) customer;
 branch.go();
 }
}

```

**Ans Option5:** An exception is thrown at runtime because (Branch)Customer is incorrect

Predict the output for the below code

```

public class Game {

 public static void main(String[] args) {
 displayRegistration("Hockey"); //Line 1
 displayRegistration("Kho-Kho", 132, 102, 36); //Line 2
 }

 public static void displayRegistration (String gameName, int... id) {
 System.out.println("Registration for " + gameName + ".");
 for(int i=0; i<id.length; i++) {
 System.out.println(id[i] + "");
 }
 }
}

```

**Ans**

```

Registration for Hockey.
Registration for Kho-Kho.
132
102
36

```

1. What is the output of the following code?

```
Class Employee {
 Void disp(char c){
 System.out.print("Employee name starts with :" +c+ ".");
 System.out.print("His experience is : 11 years. ");
 }
}

Class Main extends Employee {

 Void disp(Char c) {
 Super.disp(c);
 System.out.print("Another employee name also starts with :" +c+ ".");
 new Employee().disp("D");
 disp(7);
 }
}

String disp (int c) {
 System.out.print("His experience is :" +c+");
 return "Bye";
}
}

Public class Demo11 {
 Public static void main (String a[]){
 Employee emp = new Main();
 emp.disp("S");
 }
}
```

1. Employee name starts with : S. His experience is : 11 years. Another employee name also starts with : S. Employee name starts with : D. His experience is : 11 years. His experience is : 7.

2. Employee name starts with : S. His experience is : 11 years. Another employee name also starts with : S. His experience is 7 years
3. Employee name starts with : S. His experience is : 11 years. Another employee name also starts with : S. Employee name starts with : D. His experience is
4. Employee name starts with : S. His experience is : 11 years. Another employee name also starts with : S.

Predict the output of below code

```
public class Test {

 public static void main (String a[]){

 System.out.print("Implementing type Casting");
 Dog d = new Dog();
 BullDog bd = (BullDog) d;
 bd.show();
 }

}

class Dog{
 void show(){
System.out.print("Dog");
}
}
class Cat{
 void show(){
System.out.print("Cat");
}
}
class BullDog extends Dog{
 void show(){
System.out.print("BullDog");
}
}
```

**Ans tricky as runtime error and Implementing type casting also comes. But I guess more appropriate is runtime error**

**RUNTIME ERROR : java.lang. ClassCastException**

```
Implementing type CastingException in thread "main" java.lang.ClassCastException:
certificationJava.Dog cannot be cast to certificationJava.BullDog at
certificationJava.Dem011.main(Dem011.java:9)
```

Which code fragment can be inserted at Line 26 to display the output as “Inside catch(RuntimeException) finally end” ?

```
public class Dem011 {

 public static void main (String a[]){

 {
 try
 {
 method();
 System.out.print("Inside try");
 }
 catch (RuntimeException ex)
 {
 System.out.print("Inside catch(RuntimeException)");
 }
 catch (Exception ex1)
 {
 System.out.print("Inside catch(Exception)");
 }
 finally
 {
 System.out.print("finally");
 }
 System.out.print("end");
 }
 }
 public static void method()
 {
 //Line 26
 }
}
```

**Ans: Option A**

**throw new RuntimeException();**

Predict the output of the foll

Square.java

```
final class Square {
 private double length, breadth;
 Square(double length, double breadth) {
 this.length= length;
 this.breadth= breadth;
 }
 Square(Square square){
 System.out.println("Copy Constructor Invoked");
 length =square.length;
 breadth= square.breadth;
 }
 public String toString() {
 return "(" + length +"+"+breadth+ ")";
 }
}
```

Main.java

```
class Main{
 public static void main(String args[]){
 Square square1= new Square(110,115);
 Square square2= new Square(square1);
 System.out.println(square2);
 }
}
```

Ans: Option A

Copy Constructor Invoked  
(110.0+115.0)

Which Line fragment can be inserted in Line 1 will help to get the output as 110231(choose all apply)

```
package certificationJava;

public class InnerClassDemo {

 private int bookid=110;

 class Book
 {
 private int bookid=231;
 private int getBookid()
 {
 return bookid;
 }

 public void main (String [] args)
 {
 Book book = new Book();
 System.out.println(book.getBookid());
 }
 }

 private int getBookid() {

 return bookid;
 //Line1
 }
}
```

**Ans : Option 2 and Option 4**

```
InnerClassDemo innerClassDemo = new InnerClassDemo();
InnerClassDemo.Book book = innerClassDemo.new Book();
```

```
System.out.printf("%d",innerClassDemo.getBookid());
book.main(args);
```

and

```
InnerClassDemo innerClassDemo = new InnerClassDemo();
Book book = innerClassDemo.new Book();
System.out.printf("%d",innerClassDemo.getBookid());
book.main(args);
```

Which of the below option **fails** at Line 7(choose all that apply)

Employee.java

```
public class Employee {
 static final int empid =1101;
}

SuperDemo.class

class Unit extends Employee{
 int empid =1102;
 void display()
 {
 //Line7
 }
}
class SuperDemo {
 public static void main(String [] args)
 {
 Unit unit = new Unit();
 unit.display();
 }
}
```

## Options

- i. System.out.println("Maximum Speed"+super.empid);
- ii. System.out.println("Maximum Speed"+ new Employee().empid);
- iii. Employee emp1 = new Employee();  
System.out.println("Maximum Speed"+ new Unit().empid);
- iv. System.out.println("Maximum Speed"+ Employee.empid);

All 4 options works fine. Not sure if question is wrong or what

Given

The below code fragment can be inserted at Line 1 and Line 2.What will be the output?

```
ConstructorDemo1 constructorDemo1=new ConstructorDemo1(1101,"Jacklin");
ConstructorDemo1 constructorDemo2=new ConstructorDemo1(1102,"John",25);
```

```
class ConstructorDemo1 {
 private int id;
 private final String name;
 static final int age=22;
 ConstructorDemo1(int i,String n){
 id=i;
 name=n;
 }
 ConstructorDemo1(int i,String n,int a){
 id=i;
 name=n;
 }
 void display(){
 System.out.println(id+" "+name+" "+age);
 }
 public static void main(String args[]){
 //Line1
 //Line2
 constructorDemo1.display();
 constructorDemo2.display();
 }
}
```

```
}
```

**Ans Option B**

```
1101 Jacklin 22
1102 John 22
```

Predict the output of the below code

```
public class InnerClassDemo {
```

```
 InnerClassDemo()
 {
 System.out.print("InnerClassDemo Constructor");
 }
```

```
}
```

Demo.java

```
class Demo {

 Demo()
 {
 System.out.println("Demo Constructor");
 }
 public void disp()
 {
 System.out.print("Simple Class");
 }
}
```

```

public static void main(String[] args)
{
 InnerClassDemo innerClassDemo=new InnerClassDemo();
 innerClassDemo.createDemo();
}
void createDemo()
{
 (new Demo() {}).disp();
}

```

**Ans : Option A : Compilation fails**

Predict the output

```

class Main{

 public void display(int i) {
 System.out.println("Inside First");
 }
 public void method(int i,int j) {
 System.out.println("Inside Second");
 }
 public void method(int... k) {
 System.out.println("Inside Third");
 }

 public static void main(String args[]){
 new Main().method(110);
 }
}

```

```
 new Main().method(110,210);
 new Main().method(110,210,310); //Line1
 new Main().method(110110,210,310,410); //Line2

}
```

**Ans:**

```
Inside Third
Inside Second
Inside Third
Inside Third
```

**Predict the output**

```
public class Book {
 int bookid = 2356;
}

public class Book1 extends Book{
 int bookid = 1167;
}

public class Book2 extends Book1
{
 int bookid = 2378;
 void display()
```

```

 {
 System.out.println(super.super.bookid); //Line10
 System.out.println(super.bookid); //Line11
 System.out.println(bookid);
 }

}

class Demo {

 public static void main(String[] args)
 {
 Book2 book2 = new Book2();
 book2.display();
 }
}

```

**Ans: A. Compilation Fails because of an error in Line10**

When the following code is inserted in Line 6. Whats the output

```

Apple apple = (Apple)typeCastDemo;

class Apple {

}

public class TypeCastDemo {
 public static void main(String[] args)
 {
 TypeCastDemo typeCastDemo = new TypeCastDemo();
 }
}

```

```
//Line6
}
}
```

**Ans Option C.**

**Compilation fails as typecast cant be done from TypecastDemo to Apple**

What code fragment can be inserted at Line3 to enable the code to print 188.22

```
enum Fruits{
 APPLE,
 MANGO,
 STRAWBERRY,
```

```

LICHI;

double claculate(double a, double b) {
 switch(this) {
 case APPLE:
 return a+b;
 case MANGO:
 return a-b;
 case STRAWBERRY:
 return a*b;
 case LICHI:
 return a/b;
 default :
 throw new Assertion("Unknown input"+this);
 }
}

public class EnumDemo {
 public static void main(String[] args)
 {
 //Line3
 }
}

```

**Ans : Option A**

```
double res = Fruits.MANGO.claculate(298, 109.78);
```

Whats the output

```

public class Parent {

 void message()
 {
 System.out.println("Inside parent class");
 }
}

public class Derived extends Parent{

 void message()
 {
 System.out.println("Inside derived class");
 }

 void display()
 {
 message();
 super.message(); //Line1
 }
}

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

 Derived derived=new Derived();
 derived.display(); //Line2
 }
}

```

**Ans: Option D.**

**Inside derived class**

**Inside parent class**

What is the result of attempting to compile and run this program

```
public class Bank extends Exception{

}

public class Customer extends Bank {

}

public class ExceptionDemo {

 public static void main(String args[]) {

 try
 {
 throw new Customer();
 }

 catch (Bank bank) {

 System.out.println("Bank catch Block");
 }

 catch(Customer customer) {

 System.out.println("Customer catch Block");
 }
 }
}
```

**Ans : Option C. Compilation error because Customer class exception is not throwable.**

**Whats the output of the following**

```
public class Demo extends Book {
 int bookid =4567;

 public int getValue() {
 return bookid;
 }

 public void call() {
 System.out.println(getValue());
 System.out.println(super.getValue());
 }
}
```

```

public static void main(String args[]){
 Book book = new Book();
 book.call(); //Line3
 super.call(); //Line4
}
}

```

Book.java

```

public class Book {
 int bookid = 17897;

 public int getValue() {
 return bookid;
 }
}

```

**Ans: Option D. Compilation fails because of an error in line 3 and line 4**

2. What is the output of the following code ?

Package exceptions;

```

import java.io.*;
public class ExceptionDemo{
 static class Car implements AutoCloseable{
 public void close(){
 System.out.print("Automatic Door Close");
 }
 }
 static class carWindow implements Closerable{
 public void close(){
 System.out.print("CarWindow");
 throw new RuntimeException();
 }
 }
}

public static void main(String[] args){
 try(Car car=new Car()){
 CarWindow carWindow=new CarWindow();
 System.out.print("Inside try block");
 }
 catch(Exception e){
 System.out.print("Inside catch block");
 }
}

```

```

Finally{
 System.out.print("finally");
}
}
}
}

```

- a. Automatic Door close CarWindow Inside try block inside catch blockfinally
- b. Automatic Door Close CarWindow Inside catch blockfinally
- c. Inside try blockCarWindowAutomatic Door CloseInside catch blockfinally
- d. An exception is thrown at run time
- e. Compilation fails

**Ans : c. Inside try blockCarWindowAutomatic Door CloseInside catch blockfinally**

60. Given:

```

Public class ExceptionDemo1{
Static class Car implements AutoCloseable{
Public void close(){
System.out.print("Car door close");
Throw new RuntimeException();
}
}
Static class CarWindow implements Closeable{
Public void close(){
System.out.println("Car window close");
Throw new RuntimeException()
}
}
Public static void main(String[] args){
Try{
 //Line 1
}
Catch(Exception e){
System.out.println("Catch exception");
}
Finally{
System.out.print("finally");
}
}
}

```

Which of the below code can be inserted at Line1 to display THE OUTPUT AS “try block finally” (Choose all that apply)

A)Car car=new Car();  
**CarWindow carWindow=new CarWindow();**

**A) System.out.print("try block");**

B) Car car=new Car();  
**System.out.print("try block");**

C) Car car=new CarWindow()  
System.out.print("try block");

D) **System.out.print("try block")**

. Which two statements are true for a two-dimensional array?

- A.It is implemented as an array of the specified element type
- B.Using a row by column convention, each row of a two-dimensional array must be of same size
- C.At declaration time,the number of elements of the array in each dimension must be specified
- D.All the methods of the class Object may be invoked on the two-dimensional arrary

- a) Option (A) and (B)
- b) Option (A) and (B)
- c) Option (B) and(C)
- d) Option (C) and (D)**

Which of the below code fragment needs to be inserted at Line12 to display the output as 15.

```
public class ExceptionInClass {
 int data=10 ;
 void calculate() throws Exception
 {
 try
 {
 data++;
 try
 {
 data++;
 // Line12
 }

 catch(Exception ex)
 {
 data++;
 }
 catch(Exception ex)
 {

```

```

 data++;
 }
 }

 void display()
 {
 System.out.println(data);
 }

 public static void main(String[] args) throws Exception
 {
 ExceptionInClass exceptionInClass = new
ExceptionInClass();
 exceptionInClass.calculate();
 exceptionInClass.display();
 }

}

```

**Ans: None of the above. If not write answer as option A**

What is the output when the below code is compiled and executed?

```

import java.util.regex.Matcher;
import java.util.regex.Pattern;

public class Demo1{
 public static void main(String[] args) {
 ("x*y");
 Matcher match=pattern.matcher("y");
 Boolean boolean1=match.matches();
 System.out.println(boolean1);
 }
}

```

**Ans: Option A True**

Output of below code is

```

class Light{
 Boolean isOn;

 void turnOn(){
 isOn=true;
 }

 void turnoff(){
 isOn=false;
 }

}

```

```
class LightDemo{
 public static void main(String[]args){
 Light light1=new Light();
 Light light2=new Light();
 light1.turnOn();
 System.out.println("light1 is on");
 light1.turnOff();
 System.out.println(light1.isOn);
 System.out.println(light2.isOn);
 }
}
```

Ans: True

False

Null

What will be the output of the code given below?

```
public class ABC{

 public static void main(String[]args){
 Boolean flag=false;
 if (flag = true){
 System.out.println("true");
 if (flag = false)
 {
 System.out.println("false");
 }}}
```

Ans Option A. True

What will be the output of the following code

```
public class Test{
 public void method()
 {
 for(int i=0;i<3;i++) {
 System.out.println(i);
 }
 }
 public static void main(String[]args){
 method();
 }
}
```

```
}
```

```
}
```

**Ans: C. Compilation Fails**

What error do we get when we compile the below code

```
public class Main{

 static int[] x;
 static{
 x[0]=102;
 }
 public static void main(String[]args){
 System.out.println(x);
 }

}
```

**Ans java.lang.ExceptionInInitializerError**

What is the output for the below code

```
public class VarArgsDemo{

 static void func(int...x)
 {
 System.out.println("Number of arguments "+x.length);

 for(int i:x)
 System.out.print(i+"");
 System.out.println();
 }
 void func(int a) //Line1
 { System.out.println("one"); }

 public static void main(String[]args){
 new VarArgsDemo().func(150);
 func(11, 12, 13, 14);
 func();
 }
}
```

**Ans:Option B**

```
one
Number of arguments 4
11121314
Number of arguments 0
```

How many objects are eligible for garbage collection after executing line8.

```
public class Employee {

 double salary;
 public static void main(String[]args){
```

```

Employee employee1=null;
Employee employee2=null;
employee1= new Employee();
employee2= new Employee();
employee1= new Employee();
Employee employee3=null;
employee1= employee3=null; //Line8
System.out.println("Hello World");
}
}

```

**Ans :3 objects**

Which statements are true about the following code snippet?(choose all that apply)

```

Public class Developer{}
Public class Employee{
Public String empName;
}

Public class Tester extends Employee{
Public Developer developer;
}

Public class Testing extends Tester{}


```

a. Testing has a empName  
b. Testing has a Developer  
c. Testing is a Developer  
d. Testing is a Employee  
e. Tester is a Testing  
f. Employee has a Developer

Ans : d & a

Observe the below code snippet:

```

public class Tree extends BasePlant{

 public void growFruit(){}
 public void dropLeaves(){}

}

public class BasePlant {

 private String name;
 public BasePlant(String name){
 this.name=name;
 }

 public String getName(){
 return name;
 }

}

```

Which of the following statement is true?(choose all that apply)

- a. The code will be compiled without any changes
- b. The code will be compiled only if the below code is added to the Tree class  
`Public Tree() {super("Plant");}`
- c. The code will be compiled only if the below code is added to the BasePlant class  
`Public BasePlant() {Tree(); }`
- d. The code will be compiled only if the below code is added to the BasePlant class  
`Public BasePlant() {this("Plant"); }`

Ans : b & d

Predict the output

Apple.java

```
public class Apple {
 public void color(){
 System.out.println("Red");
 }
}
```

Mango. Java

```
public class Mango extends Apple{

 @Override
 public void color(){
 System.out.println("Yellow");
 }
 public static void main(String[] args){
 Apple apple=new Mango(); //Line1
 apple.color(); //Line2
 }
}
```

**Ans: Yellow**

Given:

```
public interface interfaceDemo{
 //Line1
}
```

Select the suitable code fragment can be inserted at Line1(choose all that apply)

- a. `void display(int x);`
- b. `void display(int x) {}`
- c. `public static void display(int x){}`
- d. `default void display(int x);`
- e. `public interface Demo{}`

**Ans : a,c and e**

Analyze the code and select the suitable outcome

```
class Apple {
 private Apple() {
 System.out.println("Apple constructor");
 }

 void display(){
 System.out.println("Apple constructor");
 }
}

public class Main {

 public static void main(String[] args){
 Apple apple=new Apple(); //Line2
 apple.display();
 }
}
```

**Ans: Option D: Unresolved compilation problem: The constructor Apple() is not visible**

Output of the below code

```
public class Demo {
 static int x=232;
 int y=135;
 public void display(){
 System.out.print("Inside Demo");}
 public static void staticMethod(){
 System.out.print(x); //Line 8
 }

 public static void main(String[] args)
 {
 Demo.staticMethod(); //Line 13
 Demo demo=new Demo();
 demo.staticMethod(); //Line15
 staticMethod();
 demo.display(); //Line 16
 }
}
```

**Ans : Option B**

**232232232Inside Demo**

**Check line13. If its not Demo.staticMethod(); then answer will be 232232Inside Demo**

Output of the below code

```
public class Demo {

 public static void main(String[] args){
 try{
 throw 110;
 }

 catch(int ex){
 System.out.println("Caught Exception" + ex);
 }
 }
}
```

```
}
```

**Ans: Compilation Fails**

Predict the output of the below

```
public class TestDemo {

 public static void main(String[] args){
 int sum, a=10, b=10;
 try{
 System.out.println(sum=a/b);
 return; //Line 1
 } catch(ArithmetricException | Exception e){ //Line2
 System.out.println(e.getMessage());
 }finally{
 System.out.println("in finally");
 }
 }
}
```

**Ans: Compilation fails because of the error in Line2**

What are the different types of memory areas used by JVM(choose two)?

1.Class

**2.Heap**

**3.Stack**

4.Queue

JVM in java is a

1.Debugger

2.Assembler

**3.compiler**

4.Interpreter

132. What is magic number in java in the context of java programming best practices?

1.A number which gets printed on the console

**2.A direct usage of the number int the code**

3.A number which magically disappears from the code

4.A number which is generated through error

What is the output of the below code

```
public class person {

 public person(String name){
 System.out.println(name);
 }

}

public class student extends person{

 public student(){ //Line 8
 System.out.println(" Student");
 }

 public static void main(String[] args) { // Line 11
 new person("Jacklin");
 }
}
```

**Ans: Compilation fails because of an error in Line 8**

3. Given:

```

public abstract class Employee {
 private int empld;
 private int salary;
 public abstract void display();
 public void setValues(int empld, int salary){
 this.empld = empld;
 this.salary = salary;
 }
}

```

Which of the following classes provide the right representation of the child class of Employee class?

- 1) public abstract class Child extends Employee {  
    private int z;  
}
- 2) public class Child implements Employee {  
    private int z;  
}
- 3) public class Child extends Employee {  
    private int z;  
    public void display();  
}
- 4) public class Child extends Employee {  
    private int z;  
    public void display() {  
        /\* code here \*/  
    }  
}

Ans : 4) public class Child extends Employee {

```

 private int z;
 public void display() {
 /* code here */
 }

```

4. Given an abstract Class Customer as below:

```

public abstract class Customer
{

```

```
public abstract String getCustomerType();
}
```

Select a Valid implementation of getCustomer Type method in another class, from the below options:

- 1) abstract class C1 extends Customer{  
 public string getCustomer Type()  
 { return "Premium";  
 }  
}
- 2) Customer customer = new Customer(){  
 public String getCustomerType()  
 { return "Premium";  
 }  
}
- 3) class C1 extends Customer{  
 public String getCustomerType()  
 { return "Premium";  
 }  
}
- 4) new Customer(){  
 public String getCustomerType()  
 { return "Premium";  
 }  
}

Ans : 3) class C1 extends Customer{

```
public String getCustomerType()
{ return "Premium";
}
}
```

Output of the following

```
class Customer{
int customerId = 11201;
Customer() {
 customerId = 11240;
}
```

```
}

class Main {
public static void main(String args[]){
 Customer customer = new Customer();
 System.out.println(customer.customerId);
}
}
```

Ans: Option B 11240

Code to be written to get the output as below

False

Simple

Demo

For

Regular

Expressions

Using

Pattern

Matching

```

public class RegExDemo {
 public static final String string1="Simple demo for +"regular expressions"+"
using pattern matching";
 public static void main(String[] args){
 //Line 1
 //Line2
 }
}

```

**Ans: Option 1**

```

System.out.println(string1.matches("\t"));
String[] splitString=(string1.split(" " + ""));
//(String1.split(\s+)) not working in my computer so did like this
for(String string: splitString){
 System.out.println(string);
}
System.out.println(string1.replaceAll("\S"," \t"));
}

```

What is the result of attempting to complete and run this program?

```

public class Demo {
 public static void main(String[] args){
 String c="a";//Line 3
 switch(c) {
 //Line4
 case 65://Line5
 System.out.println("One");
 break;
 case "a"://Line6
 System.out.println("two");
 case 3://line 10
 System.out.println("three");
 break;
 }
}

```

```
 }
}
```

**Ans:D Computation fails because of an error in Line 5 and Line 10**

Select all possible options that are valid among the following Enums can be defined inside\_\_\_\_\_

- a) An interface
- b) A class {Multiple choice question}
- c) A static Context
- d) A method

Which code fragment can be inserted at Line 1 to enable the code to print as “Number of Days =25”

```
class Demo {
```

```

public static void main(String[] args)
{
 int monthValue=2;
 int yearValue=4000;
 int numberOfDays=10;

 switch(monthValue) {

 case 1: case 3: case 5:
 case 7: case 8: case 10:
 case 12;
 numberOfDays=31;
 break;
 case 4: case 6:
 case 9: case 11:
 numberOfDays=28;
 break;
 case 2:
 //Line1

 numberOfDays=25;
 else
 numberOfDays=28;
 break;

 default:
 System.out.println("Number of Days =" +numberOfDays);

 }
}

```

**Ans: Option not clear. Either one can come. Make sure assignment operator is there(= =)**

```

if((yearValue% 4 ==0) &&
 (yearValue% 100==0)
 ||(yearValue% 400==0))

```

**Or**

```

if((yearValue% 4 ==0) ||
 (yearValue% 100==0)
 ||(yearValue% 400==0))

```

81. Identify which of the following class breaks its input into tokens using a whitespace pattern?

- a. InputStreamReader
- b. Console

- c. Scanner
- d. Buffered Reader
- e. DataInputStream

What is the output when below code is compiled and executed.

```
import java.util.regex.Pattern;
public class RegExDemo2 {
 private static final String String1="";
 private static final String String2="one two three four five";
 public static void main (String[] args){
 Pattern pattern=Pattern.compile(String1); //Line 7
 String[] strArr=pattern.split(String2); //Line 8
 for(String str:strArr){
 System.out.println(str);
 }
 }
}
```

**Ans : Option C (if pattern.compile and pattern.spilt don't have dot means then compilation error)**

C) one  
two  
three  
four  
five

What is the output of the below code

```
package certificationJava;

public class ABC {
 public static void main(String args[]){
 Boolean flag=false;
 if(flag==true){
 System.out.println("true");
 }
 if(flag==false){
 System.out.println("false");
 }
 }
}
```

**Ans: Option A . True**

Which is the correct code fragment to be inserted at Line 1 to execute the code to print count starts from 111,112,113....

```
public class Demo2 {
```

```

 public static void main(String[] args){
 int[]X={111,112,113,114,115,116,117,118,119,110};
 //Line1
 System.out.println("count is"+i);
}
}

```

**Ans : Option B**

```
for(int i:X){
```

Predict the output of the below

```

public class Calculator {

 int a=123;
 int b=200;
 public void display(){
 System.out.println("a"+a+"b"+b+"");
 }
}

public class CalculatorDemo {
public static void main(String[] args)
{
 Calculator calculator1=new Calculator(); //Line1
 Calculator calculator2= Calculator1//Line2
 calculator1.a+=1;
 calculator1.b+=1;
 System.out.println("calculator1 values");
 calculator1.display();
 System.out.println("calculator2 values");
 calculator2.display();
} }
```

**Ans : D. Compilation fails because of error in Line2**

Output of the following

```

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

 int i1=0;
 int[] j={11,111,14,19,116,215}; //line4
 for (int i1:j) //line5
 System.out.printf("%d",i1);
 }
}
```

**Ans :Option C: compilation fail because of an error in line5**

Output of the following

```
abstract class Customer {

 public int custId;
 Customer()
 {
 custId=23456;
 }
 abstract public void setId();
 abstract final public void getid(); //Line11
}

class Demo extends Customer{
 public void setId(int custId)
 {
 this.custId=custId;
 }
 final public void getid() //Line9
 {
 System.out.println("Customerid"+custId);
 }
 public static void main(String[] args)
 {
 Demo demo=new Demo();
 demo.setId(1102);
 demo.getid();
 }
}
```

Ans :

- a) compilation fails because of an error in Line9
- b) compilation fails because of an error in Line11

Output of the below code

```
public class Employee {

 public final void show(){
 System.out.println("show()inside Employee");
 }
}

final class Unit extends Employee{
 public void show1(){ //Line1
 final int x=100;
 System.out.println("show()inside Unit");
 System.out.println(x);
 }
}

public class Demo11{
 public static void main(String[] args){
 Employee employee=new Unit();
 new Unit().show1();
 }
}
```

**Ans:Option D**

**show()inside Unit  
100**

Given

```
Class Parent{
}
Class Child extends Parent{
}
Final class GrandChild extends Child{
}
```

Which of the following statement is not true about the above code?

- a) The above code represents the multi-level inheritance with the two level
- b) The GrandChild class can Access the protected and public members of the parent and child class
- c) **Instance of parent class can accept the reference of the child class but not the reference of GrandChild class**
- d) The GrandChild class can override the methods of both Parent class and Child class

In the below code snippet identify which of the following method compares the given values and return an int which tells lesser or greater

```
public class WrapperClassDemo {

 public static void main(String aa[]){
 int x=90;
 Integer i1=new Integer(x);
 int y=90;
 Integer i2=new Integer(y);

 System.out.print(i1.compareTo(i2)+" "+Integer.compare(i2,i1)+" "+i1.equals(i2)+" "+
 (i1==i2));
 }
}
```

- a) **Compare()**
- b) Equals()
- c) compareTo()
- d) ==

Output of the below code

```
public class TestDemo {
 public static void main(String[] args){
 Integer n1=new Integer(100);
 Integer n2=new Integer(100);
 Integer n3=127;
 Integer n4=127;
 Integer n5=128;
 Integer n6=128;
 int n7=129;
 int n8=129;
 System.out.print(n1==n2);
 System.out.print(n3==n4);
 System.out.print(n5==n6);
 System.out.print(n7==n8);
 }
}
```

**Ans A: false true false false**

Output of the following

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

 TreeSet<String> treeset=new TreeSet<String>();
 treeset.add("first");
 treeset.add("First");
 treeset.add("Second");
 System.out.println(treeset.ceiling("Fir"));
}
```

**Ans : Option C**

**First**

**If <String> not defined then compilation error comes**

Output of the following

```
public class TestDemo {
```

```

public static void main(String[] args){
ArrayList Strings=new ArrayList();
Strings.add("aAaA");
Strings.add("AaA");
Strings.add("aAa");
Strings.add("AAaa");
Collections.sort(Strings);
for(string:Strings){
System.out.print(string);
}}}

```

**Ans: Option A Compilation Fails**

Output of the following

```

public class person {
 private final String name;

 public person(String name){
 this.name=name;
 }

 public String toString(){
 return name;
 }
}

import java.util.TreeSet;
public class Group extends TreeSet {

 public static void main(String[] args){
 Group g=new Group();
 g.add(new person("Hans"));
 g.add(new person("Jane"));
 g.add(new person("Hans"));
 System.out.println("Total"+g.size());
 }

 public boolean add(Object o){
 System.out.println("Adding"+o);
 return super.add(o);
 }
}

```

**Ans : Option A**

a) Adding Hans

An exception is thrown at the runtime(`java.lang.ClassCastException`)

Output of the following

```

public interface StaticInterface {

```

```

static void staticMethod()
{
System.out.println("Inside interface");
}
}

public class StaticInterfaceImpl implements StaticInterface {

public void staticMethod()
{
System.out.println("Inside class");
}
}

public class StaticDemo {

public static void main(String[] args)
{
new StaticInterfaceImpl().staticMethod();
}
}

```

**Ans: Option D. code will print “inside class” on execution**

**Inside class**

Output of the following

```

import java.time.LocalDate;
import java.time.format.DateTimeFormatter;

public class Formatting {
 public static void main(String[] args)
 {
 LocalDate date=LocalDate.of(2016,11,13);

 DateTimeFormatter formatter= DateTimeFormatter.ofPattern("dd/MMM/YYYY");
 System.out.println(date.format(formatter));
 }
}

```

**Ans: Option D: 13/Nov/2016 will be printed**

**13/Nov/2016**

Predict the output

```
public interface Interface1 {
```

```

 default void method1()
 {
 System.out.println("Inside default method");
 }

}

public interface DefaultExtends extends Interface1{
 default void method1()
 {
 System.out.println("Default method redefined");
 }
}

public class InterfaceWithDefaultMethod implements DefaultExtends{
 public static void main(String[] args)
 {
 InterfaceWithDefaultMethod defaultExtend=new
 InterfaceWithDefaultMethod(); //Line4
 defaultExtend.method1(); //Line5
 }
}

```

**Ans: Option B**

**Default method redefined**

What happens when default keyword is removed from the below code snippet

```

public interface Interface1 {

 default void method1()
 {
 System.out.println("Inside default method");
 }
}

```

a.method cannot be overridden in the implementing classes

b.method can be overridden in the implementing classes

c.method cannot be given body in the interface

d.compilation error occurs

- a) a and b
- b) a,b and c
- c) c and d**
- d) b and c

Select the valid code fragment according to java coding standard?

- 1) public void draw(String s){

```

if(s.equals("Square")){
 drawSquare();
}
if(s.equals("Rectangle")){
 drawRectangle();
}
}

2) public void draw(String s){
 if("Square".equals(s)){
 drawSquare()
 }
 if("Rectangle".equals(s)){
 drawRectangle();
 }
}

```

**only option(1) is valid**

only option(2) is valid

Both(1) and (2) are valid

Both(1) and (2) are invalid

Whats the output of the below

```

public class Ex1 {
 public String formatInInput(String i){
 if(i.trim().length()==9){
 StringBuilder s1=new StringBuilder();
 s1=s1.insert(0, "+1(");
 s1=s1.insert(6, ")");
 s1=s1.insert(10, "-");
 return s1.toString();
 }
 return null;
 }
 public static void main(String args[]){
 Ex1 ob=new Ex1();
 String I;
 ob.formatInput(I);
 }
}

```

- a) compilation fails at Line3
- b) Compilation fails at Line 6
- c) Null pointer exception will be thrown if the value of I is null**

- d) Compilation fails due to error in Line7

```
// I got below compilation error
```

```
Exception in thread "main" java.lang.Error: Unresolved compilation problem:
```

Consider the below code snippet

```
Locate locate=new Locate("da","DK");
NumberFormat numberFormat=NumberFormat.getInstance(locate);
String number=numberFormat.format(100,99);
System.out.println(number);
```

Here NumberFormat.getInstance() follows which design pattern?

- a) Factory method pattern
- b) Singleton pattern
- c) Abstract Factory pattern
- d) Builder pattern

Output of the following

```
public class TestDemo {

 static int a=0;
 static ArrayList b;

 @BeforeClass
 public static void beforeClass(){
 a=10;
 b=new ArrayList();
 }
 @BeforeMethod
 public void int1(){
 a=15;
 b.add(a);
 }
 @Test
 public void test(){}
```

```

 a=a+20;
 System.out.print(a);
 System.out.println(b);
}
@Test
public void test1(){
 a=a+30;
 System.out.print(a);
 System.out.print(b);
}
}

```

Predict the output?

- a) 35[15]  
45[15,15]
- b) 35[15]  
65[15,15]
- c) 35[15]  
45[15]
- d) 35[15]  
65[15]
- e) 35[15]  
65[30]

Output of the below code snippet

```

public class Test {

 public static void main(String[] args){
 int [][] x;
 x=new int[3][4];
 for(int i=0;i<3;i+=2){
 for(int j=0;j<4;j++){
 x[i][j]=i+j;
 System.out.print(x[i][j] + " ");
 }
 }
 }
}

```

**Ans : Option B**

**01232345**

Output of the foll

```
public class Pet {
 public void displayName(){
 System.out.println("Inside Pet");
 }
}

public class Dog extends Pet{

 public void displayName(){
 System.out.println("Inside Dog");
 }
}

class Demo{
 public static void main(String[] args){
 Pet pet=new Dog();
 pet.displayName();
 }
}
```

**Ans Option D: Compilation Fails.**

Output of the following

```
public class Hello {
 public static void main(String[] args){
 String s="How\"are\"you?";
 System.out.println(s);
 }
}
```

**Ans: Option A**

**How"are"you?**

Output of the following

```
public class WrapperClassDemo {
```

```

public static void main(String args[]) {

 Integer intWrapper=Integer.valueOf("12345");
 Integer intWrapper2=Integer.valueOf("11",2);
 Integer intWrapper3=Integer.valueOf("E",16);
 System.out.println(intWrapper+" "+intWrapper2+" "+intWrapper3);
}

```

**Ans: Option C**

**12345 3 14**

Predict the output of the following

```

public class Demo11{
 public static void main(String args[]) {
 Set numbers=new HashSet();
 numbers.add(new Integer(45));
 numbers.add(88);
 numbers.add(new Integer(77));
 numbers.add(null);
 numbers.add(789L);
 Iterator iterator=numbers.iterator();
 while(iterator.hasNext())
 System.out.print(iterator.next());
 }
}

```

**Ans: Option F**

**null1789884577**

Which of the below code has to be inserted at Line1, to sort the keys in the props HashMap variable?

```

public class Demo11{
 public static void main(String args[]) {
 HashMap props=new HashMap();
 props.put("key45","some value");
 props.put("key12","some other value");
 props.put("key39","yet another value");
 Set s=props.keySet();
 //Line1
 }
}

```

**Ans: Option B. Collections.sort(s);**

Output of the following

```
public static Collection get(){
 Collection sorted=new LinkedList();
 sorted.add("B");
 sorted.add("C");
 sorted.add("A");
 return sorted;
}
public static void main(String[] args){
 for(Object obj: get()){
 System.out.print(obj+".");
 }
}
```

**Ans : Option B  
B.C.A.**

Output of the following

```
public static void main(String[] args){
 TreeSet tset=new TreeSet();
 tset.add(new item());
 TreeSet b=tset;
}
```

**Ans : Option A. Compilation Fails.**

Which of the following code snippet can be inserted at line1 to display the output as

76  
Hello

```

class Apple<A> {
 A obj;
 Apple(A obj)
 {this.obj=obj;
 }

 public A getObject()
 {return this.obj;

 }
}

class Main{
 public static void main(String[] args){
 //Line1
 }
}

```

**Ans :Option A and Option B**

```

Apple apple=new Apple(76);
System.out.println(apple.getObject());
Apple appleObj=new Apple("Hello");
System.out.println(appleObj.getObject());

```

82. Refer the below code snippets and predict the outcome?

```

Public class RepeatingAnnotations{
 @Retention(RetentionPolicy.RUNTIME)
 public @interface Chocolates{
 Favourite[] value() default();
 }

 @Favourite("Diary Milk")
 @Favourite("Kit Kat")
 @Favourite("5 star")
 @Favourite("Galaxy")
 public interface Chocolate{
 }

 @Repeatable(value=Chocolates class)
 Public @interface Favourite{
 String value();
 }

 Public static void main(String[] args){
 Favourite[] a=Chocolate.class.getAnnotationsByType(Favourite.class);
 }
}

```

```
Chocolates chocolates=Chocolate.class.getAnnotation(Chocolates.class); //Line5
for(Favourite favourite: chocolates.value()){
 System.out.println(favourite.value()); } }
```

- a. Nothing will be displayed
- b. null will be printed
- c. Runtime exception will be thrown at Line 5
- d. Dairy Milk

Kit Kat

5 Star

Galaxy

Output for the following

```
public class RepeatingAnnotations {
 @SuppressWarnings("all") //line1
 @SuppressWarnings("deprecation") //line2
 public void over()
 {
 new Date().setDate(00); }
 }
```

**Ans: Option B.**

**Compilation will not be successful as @SuppressWarnings annotation is non-repeatable in nature**

Output of the following

```
public class TestDemo {
 public static void main(String[] args){
 LocalDateTime date1=LocalDateTime.of(2017, Month.FEBRUARY,
11, 15, 30); //Line1
 LocalDateTime date2=LocalDateTime.of(2017, 2, 12, 10, 20);

 System.out.println(date1.compareTo(date2));
 }
}
```

**Ans Option A -1 will be printed as execution result**

Predict the output

```
class Apple {
 int quantity;
}

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

 Apple apple;
 System.out.println("apple quantity");

 }
}
```

**Ans: Option 5: Apple Quantity**

Output of the following

```
public class TestDemo {
 private static Object staticObject;
 public static Object createStaticObject(){
 if(staticObject==null){
 staticObject=new Object(0);
 }
 return staticObject;
 }
}
```

What changes are required in the above code for successful execution?

- 1.The method createStaticObject should be synchronized
- 2.The method createStaticObject should be private
- 3.The staticObject reference should not be static
- 4.The method createStaticObject should not return Object type

Output of the following

```
public class TestDemo {

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

```

LocalDate date=LocalDate.of(12,11,2017);
System.out.print(date);
}
}

```

Ans: Option D: Exception will be raised as date not in range

Output of the following

```

public class Demo
{
 public void division(int x,int y){
 try{
 int z=x/y;
 }
 catch(Exception e){
 System.out.print("Arithmetic Exception");
 }
 finally{
 System.out.print("finally block");
 }
 }
 public static void main(String[] args)
 {
 Demo demo=new Demo();
 demo.division(0,8);
 }
}

```

**Ans: Option 2. finally block**

Output of the below code

```

public class Demo
{
 void display()
 {
 System.out.println("x= "+x+" y= "+y);
 }

 public static void main(String[] args)
 {
 Demo thisDemo=new Demo();
 thisDemo.get().display();
 }
}

```

**Ans: C . Compilation Fails**

Output of the following

```
public class TestDemo {
 static void myCode() throws MyException{
 try{
 throw new MyException("Test exception");
 }
 catch(Error|Exception ex){
 System.out.print("Inside Error and Exception");
 }
 }

 public static void main(String[]args)throws MyException{
 try{
 myCode();
 }
 catch(Exception ex){
 System.out.print("Inside Exception");
 }
 }
}
```

**Ans Option A**

**Prints Inside Error and Exception**

Output of the following

```
public class ThisDemo
{

 int x;
 int y;
 ThisDemo(){
 x=45;
 y=56;
 }
 ThisDemo get() //Line1
 {
 return this;
 }

 void display()
 {
 System.out.printf("x=%d+y=%d",x,y);
 }

 public static void main(String[] args)
 {
 ThisDemo thisDemo=new ThisDemo();
 thisDemo.get().display();
 }
}
```

Ans: will know answer based on syso only

I got answer as  $x=^*+x+^*y=^*+y$

```
public class student {

 private School school;
 private StudentDetails stuDetails;
 private Fees fees;

 public MarksHistory marksHistory(Marks marksDetails){
 //computation
 }
}
```

**Ans: Lazy Initialization**

Output of the following

```
public class Demo11{
 public static void main(String[] args){
 Parent obj =new Child();
 obj.display();
 }
}

public class Parent {

 public void display(int a){
 System.out.println("Parent Method");
 }
}

public class Child extends Parent {

 public void display()
 { System.out.println("Child Method");
 }
}
```

**Ans: A: Compilation Fails**

Predict the output

```
public class Manager extends Employee {

 public void someManagerMethod(){
```

```

 //...
 }

public class Officer extends Employee {
{
 //...
public void someMethod(Employee e){
 Manager m=(Employee)e ; //Line 12
 m.someManagerMethod();
}
}

}

public class Demo {

 public static void main(String s){
 Officer obj=new Officer();
 obj.someMethod(new Officer()); //Line 19
 }
}

```

**Ans: Option 1:** Compilation fails because of an error in Line 12

Output of the following

```

public interface Demo1 {
 public void display(String points);

}

public class Demo2 implements Demo1{
 public void display(String points){};

}

public class Demo3 {
 public Demo1 disp(){
 return null; //more code here
 }
 public String displayValue(){ //Line6
 return null;
 //more code here
 }
}

public class Demo4 extends Demo3{
 public Demo2 disp(){
 //more code here
 return null;
 }
 private String displayValue(){
 //more code here
 }
}

```

```
}
```

Ans: Option C. compilation of class Demo4 will fail because of an error in line6

Which of the code segment is written using best practice

1. 

```
List list;
public List getList{
 if(list.size()==0)
 return null;
 else
 return list;
}
```
2. 

```
Integer i1=new Integer(11);
Integer i2=new Integer(11);
System.out.println(i1==i2);
```
3. 

```
String[] str=new String[]{"Hi","Hello","Welcome"};
List strList=Arrays.asList(str);
for(iterator itr=strList.iterator();itr.hasNext();){
 System.out.println(itr.next());
}
```

Ans : Code 2 only is valid

144. //Assume that the first two of three test cases fail in “Testclass”

// Assme all the required import statements are added

```
Public class testrunner{
 Public static void main(String [] args){
 Result result = junitcore.runclasses(testclass.class)
```

```

For (Failure failure : result.getfailures()){
 System.out.println(result.wassuccessful());
}

}

1) False
2) True
3) False false true
4) False false false

```

Output of the foll

```

public class collectionsDemo{
 public static void main(String argv[]){
 ArrayList arrList=new ArrayList();
 ArrayList arrListStr=arrList;
 ArrayList arrListBuf=arrList;
 arrListStr.add(1, "SimpleString");//line6
 StringBuffer strBuff=arrListBuf.get(0);//line7
 System.out.println(strBuff.toString());//line8
 }
}

```

**Ans: Option C.Compilation fails because of error in Line7**

Output of the following

```

public class StringTest {

 public static void main(String[] args){
 String joinString=String.join(".", "java", "programming", "course");
 String s1="JAVA", s2="java", s3="Java";
 s1.toLowerCase();
 s3=s3.replace("J", "j");
 System.out.println(joinString);
 System.out.println(s1.equals(s2)+" "+(s2==s3));
 }
}

```

**Ans Option D:**

**java.programming.course  
false, false**

### **Output of following**

```
public interface DefaultMethodInterface1 {
 default public void defaultMethod(){
 System.out.println("DefaultMethodInterface1");
 }

public interface DefaultMethodInterafce2 {
 default public void defaultMethod(){
 System.out.println("DefaultMethodInterface2");
 }

public class TestDemo implements DefaultMethodInterface1, DefaultMethodInterafce2{

 public static void main(String[] args){
 DefaultMethodInterface1 defMethln=new TestDemo();
 defMethln.defaultMethod();
 }
}
```

**Ans: Compilation fails**

. Which of these statements compile?(choose at least one)

checkbox

1. HashSet hs=new HashSet();

2. HashSet set=new HashSet();

```
3.List list=new Vector();
List values=new HasgSet();
List objects=new ArrayList();
Map hm=new HashMap();
```

#### Output of the foll

```
public class TestDemo {
 public static void main(String[] args){
 List list1=new ArrayList();
 list1.add("1");
 list1.add("2");
 list1.add("3");
 List list2=new LinkedList(list1);
 list1.add(list2);
 list2=list1.subList(2,5);
 list2.clear();
 System.out.print(list1+"");
 }
}
```

**Ans: Option 1**

**the program complies successfully and throws exception during runtime**

## Section 1:

1. Which of the following OOP terminology associated with java..... Employee has address

Ans - Inheritance

2. What is the result when the following code is compiled and executed

```
public class Test {
 Long a; //Line1
 long b;
 public Test(long c){
 b=a+c; //Line 2
 System.out.println(b);
 }
 public static void main(String[] args){
 new Test(new Long(10L));
 }
}
```

Ans: Null pointer exception in Line 2 as variable a is not

3. Given

```
class Movie implements Comparator<Integer> {
 public int comparator(Integer o1, Integer o2){
 return o2.compareTo(o1);
 }
 @Override
 public int compare(Integer o1, Integer o2) {
 // TODO Auto-generated method stub
 return 0;
 }
}
```

```

class MovieApp {
 public static void main(String[] args){
 Integer mov[] = {2019,2017,1989,1994};
 Arrays.sort(mov,new Movie());
 for (int i:mov){
 System.out.print(i+" ");
 }
 }
}

```

Ans - c

2019 2017 1989 1994

4. Identify the output:

```

public class MyDemo {
 public static void main(String[] args){
 int i =5;
 switch(i){
 case 1:
 System.out.println("One");
 break;
 case 2:
 //Line 1
 case 3:
 //Line 2
 System.out.println("Two and Three");
 case 4,5:
 //Line3
 System.out.println("Four and Five");
 break;
 default:
 System.out.println("Default");
 }
 }
}

```

Ans: Compilation error in Line 3 as multiple values are not allowed in case

5. Which of the following is correct usage of a relational operator made in if statement.

Ans : if (firstName.equals("Annie")&&salary==50000)

6. Identify the output of the below code:

```
public class TestDemo {
 public static void main(String[] args){
 boolean a = true;
 boolean b = true;
 boolean c = false;
 boolean d = true;
 System.out.println(a&&b || c&&d);
 }
}
```

Ans - true

7.

```
public class UtilTest {
 @Rule
 public ExpectedException thrown = ExpectedException.none();
 //@Test(expected = Exception.class)
 //Line1
 @Test
 public void test1() throws Exception{
 thrown.expect(NullPointerException.class);
 throw new NullPointerException();
 }
}
```

Ans - @Test

8. Which of the following component is responsible to compile, debug a java program?

Ans-JDK

9. What is the output for the below code?

```
interface Fruits{
 public void printPrice();
}
public class Apple {
 public static void main(String[] args){
 Fruits fruits = new Fruits(){
 public void printPrice(){
 }
 }
}
```

```

 System.out.println("150");
 }

};

fruits.printPrice();
}
}

```

**Ans-150**

10. Which among the following is valid option for wildcards?(select 2 options)

- A. Used to relax restriction on the variable
- B. Used in scenario where type being operated upon is not known
- C. Used in generic method type argument
- D. Can access members of super class

**Ans:**

- A. Used to relax restriction on the variable**
- B. Used in scenario where type being operated upon is not known**

11. Which of the below method name is valid as per Java naming convention?

**Ans: methodName**

12. Consider the Junit test class with junit fixture annotations and the methods as below:

```

@BeforeClass ---- init()
@AfterClass ---- close()
@Before ---- setUp()
@After ---- tearDown()
@Test----testSum1()
@Test----testEven1()

```

In which order the methods will execute?

**Ans – init() setup() testSum() tearDown() setUp() testEven() tearDown() close()**

13. Which of the following is the correct syntax to declare the abstract method evaluate?

14. Predict the output of the below code.

```

class Car{
 void start(){
 System.out.println("Car Starts");
 }
}
class Bike{
 void start(){
 System.out.println("Bike Starts");
 }
}
class Automobile extends Car{
 void start(){
 System.out.println("Automobile Starts");
 }
}
public class ExceptionDemo {
 public static void main(String[] args){
 System.out.println("Implementing Typecasting");
 Car d = new Car();
 Automobile automobile = (Automobile) d;
 automobile.start();
 }
}

```

Ans :

Displays "Implementing type casting" and RUNTIME EXCEPTION java.lang.ClassCastException

15. Analyze the below code and predict the outcome when compiled and executed?

```

public class Demo extends Book {
 int bookid =4567;

 public int getValue() {
 return bookid;
 }

 public void call() {
 System.out.println(super.getValue()); //Line 1
 }

 public static void main(String args[]){
 Book book = new Book();
 super.call(); //Line 2
 }
}

public class Book {
 int bookId = 17897;
}

```

```

 public int getValue(){
 return bookId;
 }

}

```

Ans – Compilation error in Line2 as super keyword cannot be used in static context

16. Which of the following condition will not allow the finally block to be executed?

Ans – when System.exit(1) is called

17. What is the result of the following?

```

public class TestDemo {
 public static void main(String[] args){
 try{
 throw new ArithmeticException("AE");
 }catch(ArithmaticException e2){
 System.out.println(e2.getClass());
 }catch(Exception e1){
 System.out.println("E1");
 }
 }
}

```

Ans - class java.lang.ArithmaticException

18. What is the result of attempting to compile and run this program?

```

class CustomException extends Exception{}
class Customer extends CustomException{

public class ExceptionDemo1 {
 public static void main(String[] args){
 try{
 throw new Customer();
 }catch (CustomException customException){
 System.out.println("Custom Exception Catch block");
 }catch(Customer customer){
 System.out.println("Customer catch block");
 }
 }
}

```

Ans – Compilation error because customer class exception is not throwable

19. Which of this statement is not correct and will lead to compilation error.....

20. What will be the output of the following code when executed?

```
public class DateTimeTester {
 public static void main(String[] args){
 LocalDateTime localDateTime = LocalDateTime.of(2020,5, 13, 20, 46);
 System.out.println(localDateTime.get(ChronoField.HOUR_OF_DAY)
+localDateTime.getDayOfMonth());
 }
}
```

Ans – 33

21. Which of the below code is implemented without best practices standard?

1. String[] str=new String[]{"Hi","Hello","Welcome"};  
List strList=Arrays.asList(str);  
for(iterator itr=strList.iterator();itr.hasNext();){  
System.out.println(itr.next);
  
2. Integer i1=new Integer(11);  
Integer i2=new Integer(11);  
System.out.println(i1==i2);

Ans: Option 1 doesnot follow best practices. Can be improved using for(String s:str)

22. Which of the following is used for the automatic accurate tracking for the decimal values:

Ans:BigDecimal

23. Given:

```
public class TestDemo1 {
 public static void main(String[] args)
 {
 int i=4;
 int j=4;
 System.out.println(i==j);
 Integer w1=new Integer(4);
 Integer w2=new Integer(4);
 System.out.println(w1==w2);
 }
}
```

Ans: no issues in the above code

24. Consider the following statements:

1.

Interface:

1) Consider the below code snippet:

```
interface Student {
 int student_id=101;
}
class Studentimpl implements Student{
 void getStudentId() {
 student_id=102;
 }
}
```

Compilation failed  
file.java:7: error: cannot assign a  
value to final variable student\_id  
student\_id=102;

What will happen when the above code compiles?

- a) Compilation Error as student\_id is static field and hence we cant change its value after initialization.
- b) The code will compile successfully.
- c) The code will compile successfully but when executed , it will lead to runtime exception as student\_id field cannot be changed.
- d) Compilation error as student\_id will not be visible in StudentImpl

Answer : A

70) What is the output of the below code snippet?

Interface ParentInterface

```
{
 Default void display()
 {
 System.out.println("Inside default method");
 }
}
```

Interface childInterface

```
{
 Default void display()
 {
 System.out.println("Default method redefined");
 }
}
```

```

}

Public class Demo implements ParentInterface, childInterface{

 Public static void main(String[] args)

 {

 Demo demo = new Demo();

 Demo.display();

 }

}

<<<No Options>>>

```

**Answer:**

Duplicate default method display from parent AND child interface

**Objects:**

2) Consider the below code snippet:

```

public class TestDemo {

 public static void main(String[] args) {
 // TODO Auto-generated method stub
 int i=4;
 int j=4;
 System.out.println(i==j);
 Integer k=new Integer(4);
 Integer l=new Integer(4);
 System.out.println(k==l);
 }
}

```

True False

What would be the issue in the above code?

- a) Comparing Wrapper classes objects using == is wrong as it checks if both points to same object
- b) Comparing primitive data types using == is wrong
- c) Both A and B
- d) No issues in the above code

**Answer : A**

**Date and Time:**

1) Which will be the output of the code given below?

```
public static void main(String[] args){
 LocalDateTime date1 = LocalDateTime.now();
 System.out.println(date1.plus(Period.ofDays(-1)));
```

2020-08-03T14:27:42.015

- a) Yesterday's Date and Time
- b) Error as LocalDateTime.now() method is not defined
- c) Will print nothing as date1.plus() method has void as its return type
- d) Error as Period.ofDays() method only take positive values

Answer : A

2) Given

```
public class Sample {
 public static void main(String[] args) {
 LocalDateTime dateTime = LocalDateTime.of(2020, 06, 7, 1, 1).plusHours(30);
 DateTimeFormatter formatterPattern = DateTimeFormatter.ofPattern("dd-MMM-yyyy")
 String str = dateTime.format(formatterPattern)
 System.out.println(str);
 }
}
```

Predict the output?

- a) 08 -06-2020
- b) 07-Jun-2020
- c) 08-Jun-2020
- d) 09-06-2020

Answer : C

3)

```
LocalDateTime localDateTime=LocalDateTime.of(2020, 5, 13, 20, 46);
System.out.println(localDateTime.get(ChronoField.HOUR_OF_DAY)+localDateTime.getDayOfMonth());
```

Ans: 33

Annotations:

- 1) Refer the below code snippet and predict the outcome:

```
import java.lang.annotation.Repeatable;
```

```

import java.lang.annotation.Retention;
import java.lang.annotation.RetentionPolicy;
class annotationdemo {

 @Retention(RetentionPolicy.RUNTIME)
 public @interface Chocolates{
 Favourite[] value() default{};
 }

 @Repeatable(value = Chocolates.class)
 public @interface Favourite{
 String value();
 }

 @Favourite("Dairy Milk")
 @Favourite("Kitkat")
 @Favourite("5 Star")
 @Favourite("Galaxy")
 public interface Chocolate{

 }
}

```

Dairy Milk  
Kitkat  
5 Star  
Galaxy

```

public static void main(String args[]){
 Chocolates Chocolate = Chocolate.class.getAnnotation(Chocolates.class);
 for(Favourite favourite:Chocolate.value()){
 System.out.println(favourite.value());
 }
}

```

### Arrays:

#### 1) Given

```

public class ArraysDemo{
 public static void main(String[] args){
 int[] arrVar={11,22,33,44,55,66,77,88,99,109};
 }
}

```

```

int position=3;
int value=7;
System.out.println("Original Array : "+Arrays.toString(arrVar));
for(int i=arrVar.length-1;i>position;i--){
 arrVar[i]=arrVar[i-1];
}
arrVar[position]=value;
System.out.println("New Array: "+Arrays.toString(arrVar));
}

```

Identify the outcome of the given code snippet

**ANSWER :**

New Array: [11, 22, 33, 7, 44, 55, 66, 77, 88, 99]

Original Array : [11, 22, 33, 44, 55, 66, 77, 88, 99, 109]  
 New Array: [11, 22, 33, 7, 44, 55, 66, 77, 88, 99]

2) class Demo1 {  
 public static void main(String[] args) {  
 int i=0;  
 int[] myArray = {11,111};  
 for(int i : myArray) { //Line1  
 System.out.println("%d",i);  
 }}}

- a) Primitive types cant be used in an advance for loop.
- b) Variable i cannot be resolved to a type
- c) Duplicate Local Variable i
- d) Type mismatch: cannot convert from element type int to i

**Answer : c**

3) what is the output of the following code?

Given:

```
public class Main {
```

```

public static void main(String[] args) {

 int twoD[][]=int int[4][]; //Line1
 twoD[0] = new int[1];
 twoD[1] = new int[2];
 twoD[2] = new int[3];
 twoD[3] = new int[4];

 for(int i = 0;i<4;i++){
 for(int j=0;j<i+1;j++){
 twoD[i][j]; //Line 2
 }
 }

 System.out.println("executed");}
}

```

**Answer :-**~~Executed~~ compilation error at line 2-

4) 38) Given

```

Class ArrayDemo{

public static void main(String[] args){

 int x[] = display();

 for (int i=0; i< x.length; i++)
 System.out.print(x[i]+” “);

}

Public static int[] display(){

//Line1
}
}

```

Which code fragment can be inserted at Line1 to print the output as 112 142 213?

- a) new int[](112,142,213);
- b) new int(112,142,213);
- c) return new int({112,142,213});
- d) return new int[](112,142,213);

**Answer:** D ~~return new int[] {112, 142, 213};~~

60) What is the output of the following code?

```

Public class Demo11{

Public static void main(String args[]){
}
}

```

```

Int x[][] = new int[4][]; //Line1
X[0] = new int[1];
X[1] = new int[2];
X[2] = new int[3];
X[3] = new int[4];

Int a,b,c = 0;
For(a=0;a<4;a++){
 For(b=0;b<a+1;b++){
 X[a][b]=c;
 System.out.print(x[a][b] + " ");
 }
}

```

**Ans: 0 1 2 3 4 5 6 7 8 9**

**0 1 2 3 4 5 6 7 8 9**

### Collections:

1) int[] myArray = new int[] {1,2,3,4,5}

Which code snippet given below allow you to create a list from this array?

- a) List myList = myArray.asList();
- b) List myList = Arrays.asList(myArray);
- c) List myList = new ArrayList(myArray);
- d) List myList = Collections.fromArray(myArray);

**Answer: B**

2) Consider the below code snippet

```
String[] customers = new String[]{"John","Jack","Jacklin","Jasmine"};
```

```
List customersList = Arrays.asList(customers);
```

Best Code to iterate through the customersList obj data

i. for(Iterator itr = customersList.iterator();itr.hasNext();){

```

System.out.println(itr.next());
}

ii. for(String s:customerList)
System.out.println(s);

iii. for(Iterator itr = customersList.iterator();itr.next();){
System.out.println(itr.next());
}

iv) for(Iterator itr=customersList.iterator(customersList.size());itr.hasPrevious();){
System.out.printl(s.previous());
}

a) Option i
b) Option ii
c) Option iii
d) Option iv

```

**Answer : B A**

3) Assuming all the necessary imports are done , what will be the output of the following code snippet?

```

class Movie {

private String movieName;

public Movie(String Name) {

this.movieName =name;

}

@Override

public boolean equals(Object obj) {

return true;}}
```

```

public class CollectionTester{

public static void main(String[] args) {

HashMap<Movie,String> haspMap = new HashMap<>();

hashMap.put(new Movie("Harry Potter1"),"movie1");

```

```
hashMap.put(new Movie("Harry Potter1"),"movie2");
hashMap.put(new Movie("Harry Potter1"),"movie3");
System.out.println(hashMap.size());
System.out.println(hashMap.get(new movie("Harry Potter")));
```

Answer : 3 Null

4) Given

```
class Task {
```

```
 public static void main(String[] args) {
 TreeSet set = new TreeSet();
 set.add("a");
 set.add("6");
 set.add("c");
 Iterator itr = (Iterator)set.iterator();
 while(itr.hasNext()){
 System.out.println(itr.next()+"");}
 }
```

Predict the Output?

- a) a 6 c
- b) a followed by exception
- c) 6 a c

Answer : a - c

5) 37) Given:

```
Public class Demo 11 {
```

```
 Public static void main{String args[]} {
```

1. Set numbers = new HashSet();
2. numbers.add(new Integer(45));
3. numbers.add(88);
4. numbers.add(null);
5. numbers.add(789L);
6. for (Object object : numbers) {
7. System.out.print(object);

8.)

Which of the following statement is true? **No Answer.** **null 789 88 45**

6) Assuming all the necessary imports are done, what will the output when the below code gets executed?

```
Public static Iterator getIterator(List list){
 Collection.rotate(list,1);
 Collection.reverse(list);
 Return.list.iterator();
}

Public static void main(String[] args){
 List list = new ArrayList<>();
 List.add(404);
 List.add(390);
 List.add(503);
 Iterator iterator=getIterator(list);
 While(iterator.hasNext())
 System.out.println(iterator.next()+" ");
}
```

**Ans:**

**Option A:**

**390**

**404**

**503**

58) Predict the output of the below code snippet?

```
ArrayList list = new ArrayList();
List.add("Infosys");
List.add("Google");
For(String s:list){
```

```
System.out.print(" "+s);
List.clear();
}

Option A: It prints Infosys
Option B: Compilation fails as the line "for(String s:list)" cannot convert from elementtype
Ans: Option B
```

### Assertion:

14) Given

```
public class AppTest {

String message ="Hello";
int length = message.length();

@Test
public void testOne(){

System.out.print(length + " ");
assertEquals(length,5);

}

@Test
public void testTwo(){

System.out.print(length + " ");
assertEquals(length,5);

}

@After
public void teardown() {

length = length +1;

System.out.print(length +");}

}}
```

What is the result?

Answer : Both test will pass and print 5 6 5 6 in the console

2) Predict the output of the following code?

```
//Assume all the required imports are added
public class TestDemo{
 @Test
 public void test() {
 String a = “ ”;
 Assert.assertNotNull(a);}}}
```

- a) Test Passes
- b) Test fails
- c) An exception is thrown at runtime
- d) Compilation fails

**Answer : A**

77. //Assume all the required imports are added

```
public class TestDemo{
 String a1[] = { “one”, “Two”, “three” };
 String a2[] = { “one”, “Two”, “three” };

 @Test
 public void test(){
 // Line 1
 }
}
```

Choose the wrong option?

**Answer: If we place Assert.assertSame(a1,a2): at Line1 the test case will pass as it verifies the contents**

### Exceptions Handling:

1) What is the output for the below code?

```
//Myexception is custom exception class
public class TestDemo{
 static void myCode() throws MyException {
 try{
```

```

 throw new MyException("TestException");

 } catch (Error | Exception ex) {

 System.out.print("Inside Error and Exception")}

 public static void main(String[] args) throws MyException {

 try {

 myCode();

 } catch(Exception ex){

 System.out.print("Inside Exception");}}}

a) prints "Inside Error and Exception"
b) An exception is thrown at runtime
c) Prints "inside Exception"

```

**Answer : A**

2) what is the output of the following code?

package exceptions;

public class ExceptionDemo {

```

static class Car implements AutoCloseable {
 public void close() {
 System.out.print("Automatic Door close");
 }
}
public static void main(String[] args){
try{
 Car car = new Car();
 System.out.print("inside try block");
}
catch(Exception e){
 System.out.print("Inside catch block");
}
finally{
 System.out.print("finally");
}}}

```

}

**Answer : Automatic Door Close inside try block finally**

**inside try blockfinally**

3) Given

```
public class Sample {
 public static void main(String[] args) throws Exception {
 try {
 System.out.println("In try block");
 System.exit(0);
 } catch(Exception ex){
 System.out.println("In catch block");
 ex.printStackTrace();
 } finally{
 System.out.println("In finally block");
 }
 }
}
```

Predict the Output

- a) In try block In finally block
- b) In try block

**Answer : B**

28) What is the result when the following code snippet is compiled?

```
class Tester {
 public static void main(String[] args) throws Exception {
 try {
 throw new IOException(); //Line1
 } catch(IOException | Exception e) { //Line2
 System.out.println(e + "caught"); //Line3
 }
 }
}
```

- a) The code will compile and prints IOException caught
- b) The code will not compile due to error in Line 2 i.e IOException already caught.
- c) The code will not compile and shows compile time error in Line 2 and Line 3
- d) The code will compile and prints java.IOException caught

**Answer :- A B**

## String Functions:

```
1)public class Util {
 public static void main(String[] args){
 String name = "Martin";
 Name.concat="Luther";
 System.out.println(name.concat(" King"));}
}
```

Predict the output:

- a) Martin
- b) Martin Luther King
- c) Martin Luther
- d) Martin King

**Answer : D**

## Child class, super keyword and Extends:

1) Analyze the below code and predict the outcome when compiled and executed?

```
public class Demo extends Book {
 int bookid=4567;
 public int getValue() {
 return bookid;
 }
 public void call() {
 System.out.print(super.getValue()); //Line 1
 }
 Public static void main(String[] args) {
 Book book = new Book();
 Super.call(); //Line2
 }
 class Book {
 int bookid =17897;
 public int getValue() {
```

```
return bookid;}}
```

Answer : 17897 **Compilation error at line 2**

2) What is the output of the following code?

```
class Employee {
 void display(char c) {
 System.out.println("Employee name starts with : "+c+");
 System.out.println("His Experience is 11 years")};

 class Main extends Employee {
 void display(char c) {
 super.display(c);
 System.out.println("Another employee name also starts with: "+c+");
 new Employee().display("D");
 display(7);
 }
 String display(int c){
 System.out.println("His experience is : +c" years);
 return "Bye";};
 }
}

No Answer

added code:


```
class Parent {  
  
    void display(char c) {  
        System.out.println("Employee name starts with : "+c+ " ");  
        System.out.println("His Experience is 11 years");  
    }  
}  
  
public class ClassDemo2 extends Parent {  
  
    void display(char c) {
```


```

```
super.display(c);

System.out.println("Another employee name also starts with: "+c+" ");

new Parent().display('D');

display(7);

}

String display(int c){

System.out.println("His experience is : "+c+" years");

return "Bye";

}

public static void main(String []args) {

ClassDemo2 obj=new ClassDemo2();

obj.display('e');

System.out.println(obj.display(4));

}

}
```

Option A:

His experience is: 7 Years

His experience is: 11 years

Option B:

Employee name starts with: S

His experience is: 11 years

Another employee name also starts with: S

His experience is: 7 years

Option C:

Employee name starts with: S

His experience is: 11 years

Another employee name also starts with: S Employee

His experience is: 11 years

His experience is: 7 years

Option D:

Employee name starts with: S

His experience is: 11 years

Another employee name also starts with: S

Employee name starts with: D

Ans:

Employee name starts with : e

His Experience is 11 years

Another employee name also starts with: e

Employee name starts with : D

His Experience is 11 years

His experience is : 7 years

His experience is : 4 years

Bye

3) what is wrong with respect to the code that is given below

```
class Student {
 protected static String mailDomain = "@infosys.com";
 //instance variables and methods follow
}

class Hostelite extends Student{
 public void generatedReport(String name)
 {
 System.out.println(name+Student.mailDomain);}
}
```

- a) The code will not compile as the static variables of a class cannot be used inside the instance specific methods of the child class.
- b) The Code will compile but will not get executed as the static variables of a class cannot be used inside the instance specific methods of the child class
- c) The code seems to be perfect and will work as expected
- d) The code will not compile as the keyword, implements is not used for inheriting from the parent class

**Answer : C**

```
4) 39)public class Project{
 Private Integer projectId;
 Private String projectName;
 Public static void main(String[] args){
 Project oldProject=null;
 Project newProject=null;
 oldProject=new Project();
 newProject=new Project();
 newProject=oldProject;
 oldProject=new Project();
 }
}
```

Which of the following statement is true after the code gets executed?

- a) oldProject and newProject are referring to different objects.
- b) oldProject and newProject are referring to same objects.
- c) oldProject is referring to two different objects at the same time.
- d) newProject is referring to three different objects at the same time.

**Answer: A**

42) what is the result when the following code is compiled and executed?

```
Class Demo{
```

```
int x = 1;
int y = 2;
```

```
Demo display(Demo demoParam)
{
 Demo obj=new Demo();
 obj=demoParam;
 obj.x=demoParam.x++ + ++demoParam.y;
 demoParam.y=demoParam.y;
 return obj;
}
```

```
Public static void main(String[] args){
```

```
Demo obj1=new Demo();
Demo obj2=obj1.display(obj1);
```

```
System.out.println("obj1.x = " + obj1.x + "obj1.y = " + obj1.y);
System.out.println("obj2.x = " + obj2.x + "obj2.y = " + obj2.y);
}
```

- a) obj1.x=4 obj1.y=4  
obj2.x=4 obj1.y=4
- b) obj1.x=3 obj1.y=3  
obj2.x=4 obj1.y=3
- c) obj1.x=4 obj1.y=3  
obj2.x=4 obj1.y=3
- d) obj1.x=3 obj1.y=4  
obj2.x=4 obj1.y=3

Answer: No Answer

**obj1.x = 4obj1.y = 3  
obj2.x = 4obj1.y = 3**

57) What is the result of the following code?

```
Public class Vehicle{
```

```
 Static class Car{
```

```
 Public void go(){
```

```
 System.out.println("Car Ignition");
```

```
}
```

```
}
```

```
Static class ElectricCar extends Car{
```

```
 Public void go(){
```

```
 System.out.println("ElectricCar Ignition");
```

```
}
```

```
}
```

```
Static class PetrolCar extends Car{
```

```
 Public void go(){
```

```
 System.out.println("PetrolCar Ignition");
```

```
}
```

```
}
```

```
Public static void main(String[] args){
```

```
 Car car = new ElectricCar();
```

```
 Car.go();
```

}

<<<No Options>>>

Yes, you can **declare a class static** in Java, provided the **class** is inside a top-level **class**. Such clauses are also known as nested **classes** and they can be **declared static**, but if you are thinking to make a top-level **class static** in Java, then it's not **allowed**

65) What is the result when the following code is compiled and executed?

Public class Test{

```
 Public void method(){
 For(int i=0;i<3;i++){
 System.out.print(i);
 }
 }
```

Public static void main(String args[]){

```
 Method();
}
```

Option A: 012

Option B: 0 1 2 3

Option C: Compilation fails as cannot make a static reference to the non static method

Option D: 2 1 0

Ans: Option A- Option C:

68) Predict the output of the below code

Class Car{

```
 Void start(){
 System.out.println("car Starts");
 }
}
```

Class Bike{

```

Void start(){
 System.out.println("Bike Starts");
}

Class Automobile extends Car{

 Void start(){
 System.out.println("Automobile Starts");
 }
}

Public class ExceptionDemo{

 Public static void main(String[] args){
 System.out.println("implementing type casting");
 Car d = new Car();
 Automobile automobile = (Automobile)d;
 }
}

```

**Answer:**

```

implementing type casting

Exception in thread "main" java.lang.ClassCastException: snippet.Car incompatible
with snippet.Automobile

at snippet.class68.main(class68.java:22)

```

64) Given

```

Class Aircraft{

 String name = "MiG";

 String start(){

 Return "main engine Start";
 }
}

Class CivilianAircraft extends Aircraft{

 String name = super.name;
}

```

```

String start(){ //Line 1
 Return "Secondary engine start";
}
}

Public class Demo{
 Public static void main(String[] args){
 New Demo().go();
 }
}

Void go(){
 Aircraft aircraft = new CivilianAircraft(); //Line2
 System.out.println(aircraft.name + "" + aircraft.start());
}
}

```

Option A: MiG Main engine start

Option B: MiG Secondary engine start

**Ans: Option B**

Pattern:

1) What changes need to be made in the following code to make the singleton pattern correct?(Choose any 2)

```

public class Employee {
 public static Employee employeeInstance;
 private Employee() {}
 public static Employee getEmployee()
 {
 if(employeeInstance==null){
 employeeInstance = new Employee();
 }
 }
}
```

- ```
return employeeInstance;}}
```
- a) None of the Singleton Pattern is properly implemented
 - b) Rename employeeInstance to Instance
 - c) Add synchronized to getEmployee()
 - d) Change the access modifier of employeeInstance from public to private

Operations:

```
1) public class TestDemo {  
    public void main(int x) {  
        System.out.println("Main1")  
    }  
    public static void main(String args[]){  
        System.out.println("Hello Main");}  
}
```

- a) Main1
- Hello Main
- b)Hello Main
- Main1
- c) Main1
- d)Hello Main

Answer : D

44) Predict the output of the following code:

```
Public Class Main{  
    Public void display(int i)  
    {  
        System.out.println("inside first");  
    }  
    Public void method(int i, int j)  
    {  
        System.out.println("inside second");  
    }  
    Public void method(int...k)  
    {  
        System.out.println("inside third");  
    }  
    Public static void main(String[] args){  
    }
```

```
new Main().method(110);
new Main().method(110,210);
new Main().method(110,210,310); //Line1
new Main().method(110,210,310,410); //Line2
}}
```

- a) inside first
 inside second
 inside third
- b) inside first
 inside second
 inside third
 inside third
- c) inside third
 inside second
 inside third
- d) inside third
 inside second
 inside third
 inside third

Answer: B

45) What is the result when the following code is compiled and executed?

```
Public class Test{
    Public void method(){
        for(i = 0; i<3; i++){
            System.out.print(i);
        }
    }
    Public static void main(String[] args){
        method();
    }
}
```

- a) 012
- b) 0 1 2 3
- c) Compilation fails as cannot make a static reference to the non static method.
- d) 2 1 0

Answer: C

59) What is the output for the below code snippet?

```
Public class TestDemo{
```

```

Public static void main(String[] args){

    Try{
        Int a = 20/20;

    Int b = 20/10;

    System.out.println("a="+a+"b="+b);

    Try{
        If(a==1){

            A=a/(a-a);

        }

        If(b==2){

            Int c[]={1};

            C[22]=99;

        }

    }

    Catch(ArithmeticeException ae){

        System.out.println("ArithmeticeException Bock 1");

    }catch(ArrayOutofBoundsException ai){

        System.out.println("ArrayIndexOutofBoundsException Block");

    }

    Catch(ArithmeticeException ae){

        System.out.println("ArithmeticeException Block 2");

    }

}

```

Answer:

```

a=1b=2
ArithmeticeException Bock 1

```

3)

```
public class OperatorsDemo {
```

```

public static void main(String[] args) {
    // TODO Auto-generated method stub

    int x=120,y=110;

    String s1="Thank",s2="Thank";

    int arr1[]={1,2,3};

    int arr2[]={1,2,3};

    boolean boo=true;

    System.out.println("X==y:"+ (x==y));

    System.out.println("X<=y:"+ (x<=y));

    System.out.println("s1==s2:"+ (arr1==arr2));

    System.out.println("boo=true:"+ (boo==true));

}

}

```

Ans:

X==y:false

X<=y:false

s1==s2:false

boo=true:true

4) correct usage of relational operator inside if statement:

String firstName = "Annie";

int salary = 0;

- b) if(firstName=="Annie");
- c) if(firstName.equals("Annie"));
- d) if(firstName.equals("Annie")&& salary==50000);**
- e) if(firstName=="Annie" | !salary==50000);

6) int x1=5;

```

int y1=7;

System.out.println("~x1=~x1);//line1

x1&=y1;

```

```
System.out.println("x1="+x1);
```

Ans:

x1=-6

x1=5

Sorting:

53) Predict the output of the below code snippet?

```
Collection sorted = new LinkedList();
```

```
Sorted.add("B");
```

```
Sorted.add("C");
```

```
Sorted.add("A");
```

```
For(Object object : sorted){
```

```
    System.out.print(object +",");
```

```
}
```

Option A: A,B,C

Option B: B,C,A

Ans: Option B

Static Block:

78. What will be written at Line 1 so that the below code will compile and run successfully?

```
Public class Main {
```

```
    // line 1
```

```
    static {
```

```
        X[0]=102;
```

```
    } public static void main( String[] args) {
```

```
        System.out.println(x[0]);
```

```
}
```

Ans : static int[] x = new int[3]

Unit Test:

```
71) public class TestDemo{  
    @Before  
    Public void beforeTest1(){  
        System.out.println("in before test2");  
    }  
    @Before  
    Public void beforeTest2(){  
        System.out.println("in before test 1");  
    }  
    @Test  
    Public void test(){  
        String a = "123";  
        Assert.assertSame("123,a);  
    }  
}
```

Answer:

```
in before test 1  
in before test2
```

Constructor:

```
1) class Greet{  
    private static Greet greet=new Greet();  
    private Greet() {  
    }  
    public static Greet getInstance() {  
        return greet;  
    }
```

```

public void displayMessage() {
    System.out.println("Hey! have a great day!");
}

}

public class Tester {
    public static void main(String[] args) {
        Greet greet=new Greet();
        greet.displayMessage();
    }
}

```

}

Ans:

Compilation error: Constructor Greet() is not visible

Regex:

```

1) import java.util.regex.Matcher;
import java.util.regex.Pattern;

```

```

public class TestDemo {

```

```

    public static void main(String[] args) {
        // TODO Auto-generated method stub
    try {
        checkName("TioT");
    }catch(Exception r) {
        System.out.println("Exception 1");
    }
}

```

```

}

static void checkName(String name) {
    Pattern regex=Pattern.compile("T[aei]{3}T");
    Matcher mobileMatcher=regex.matcher(name);
    try {
        if(mobileMatcher.matches()==false) {
            throw new RuntimeException("Name is invalid");
        }
    }else {
        System.out.println("Name is valid");
    }
}catch(NullPointerException r) {
    System.out.println("Exception 2");
}
}
}

```

Ans: Exception 1

Patterns:

- 1) Which of the below statement indicate the need to use of Factory pattern?
 - a) when we want two classes to use the same object
 - b) we only want one instance of the object to exist
 - c) we want to build a chain of objects
 - d) when we want to allows the sub-classes to choose the type of objects to create

Answer : D

- 2) Which among the following is/are true about the design pattern? Select any two option
 - a) Design pattern depends upon abstraction
 - b) Design patterns are completed designs that can be transformed directly into code.

- c) Design pattern depends on abstraction, follows the process of dependency injection.
- d) Design pattern is a template of solving problem that can be used in many real world software development problems.

Answer: A & B.

63) Which of the following pattern refers to creating duplicate object while keeping performance in mind?

- Option A: Builder Pattern
- Option B: Bridge Pattern
- Option C: Prototype Pattern
- Option D: Filter Pattern

Ans: Option C

Collections:

- 1) which of the following statements are true if a duplicate element objT is added to a HashSet?
- a) The element objT is not added and add() method returns false.
 - b) the element objT is added successfully
 - c) An exception occurs during runtime.
 - d) An exception occurs during compile time.

Answer: a

46. Which datatypes can be stored by Java collections?

- A. Boolean
- B. int
- C. Byte
- D. Character

Primitive data types cannot be stored in Java collections

Which are boolean , byte , char , short , int , long , float and double

Answer : D

47.Which among the following option are correct with respect to HashMap?

- A. override boolean equals(Object o)
- B. override toString()

- C. override hashCode()
- D. override String hashCode()

Answer : C

Coding Standards:

1) Which of the below code is implemented without best practice standard

i) `String[] str = new String[]{"Hi", "Hello", "Welcome"};`

```
List strList = Arrays.asList(str)
```

```
For(Iterator itr = strList.iterator();itr.hasNext();){
```

```
System.out.println(itr.next());}
```

ii) `Integer i1 = new Integer(11);`

```
Integer i2 = new Integer(11);
```

```
System.out.println(i1==i2)
```

- a) Option(i) doesn't follow best practices can be improved using `for (String s: strList)`
- b) Option(ii) doesn't follow best practices as objects should not be compared with ==
- c) Option(i) and (ii) are not following best practices.
- d) Option(i) and (ii) are following best practices.

Answer : C

2) Which of the below method is valid as per java naming convention?

- a) METHOD_NAME
- b) MethodName
- c) methodName
- d) Method_Name

Answer : C

String Literals:

9) Which of the below are not good practice for creating objects?

- a) Lazy initialization of objects
- b) Creating String literals instead of String objects**
- c).Creating Wrapper objects instead of primitives
- d).Invoking static factory methods for immutable classes

Ans : B

Final:

1) Which of the following keyword is used to prevent the content of a variable from being modified from outside?

- a) final
- b) constant
- c) static
- d) transient

Answer : a

Enum:

17) Which of the following are true about enums in java?(Choose any 3)

- a) Enums can implement any other interface in java
- b) An instance of enum can be created outside of enum itself
- c) Enums cant extend any other class except the abstract base class java.lang.enum
- d) Enums can be used to implement Singleton design pattern

Answer : a , c , d

Abstract Class:

19) Which of the following statement regarding an abstract class are true in java?

- i) Object of an abstract class cant be created
 - ii) An abstract class is designed only to act as a base class in hierarchy to be inherited by other classes.
-
- a) Only i
 - b) Only ii
 - c) Both I and ii
 - d) Neither I and ii

Answer : B

35) which of the following mechanism in OOP is a process of hiding certain details and showing only essential information to the user?

- a) Polymorphism
- b) Encapsulation
- c) Abstraction

d) Inheritance

Answer : C

62) If the Child class of an abstract class does not implement all its abstract methods then it should be declared as?

Option A: Abstract class

Option B: Non Abstract class

Option C: Static class

Option D: Interface

Ans: Option A

Interface:

1) which of the following interface should be implemented by a resource, if we want to create the object of that resource in the resource part of a try block?

a) Closeable

b) Closeable<E>

c) AutoCloseable

d) AutoCloseable<E>

Answer : C

41) which of the following statements are FALSE?

(Choose 2 options)

a) An Interface can extend from only one interface.

b) A class can extend from another class and at the same time implement any number of interfaces.

c) A class can extend multiple abstract classes.

d) Many classes can implement the same Interface.

Answer: A, C

Vargs:

48. How many numbers of values can be accommodated by Varargs in Java?

A. Atleast one values

B. Atmost two values

C. More than one value

D. Any number of values

Answer: D

75. At which position should Varargs be placed in a parameterized method?

- A.First place
- B.Last Place
- C.Second Last place
- D.Can be anywhere

Ans : B

Inheritance:

49. Which of the following keyword can be used to restrict a class to be inherited in Java?

- A.Abstract
- B.final
- C.constant
- D.private

Answer :-D-B

Type Conversion:

52) Which of the following is a necessary condition for implicit type conversion in Java?

- Option A: The destination type is smaller than source type
- Option B: The destination type is larger than source type
- Option C: The destination type can be larger or smaller than source type
- Option D: The destination and the source type are the same.

Ans: Option B

Loops:

54) Which of the following Jump statement can skip processing of one iteration if a specified condition occurs and remaining iterations?

- Option A: break
- Option B: return
- Option C: continue

Option D: exit

Ans: Option C

Constructors:

55) Which of the following is false regarding parameterized constructors in Java?

Option A: Parameterised constructors should have void as return type

Option B: Parameterised constructors can take any number of parameters

Option C: Parametersied constructors cannot have private access modifier

Option D: Parameterised constructors cannot throw an exception

Ans: Option A

74. What is the correct way of placing “this” keyword in a contructor?

A.First statement

B.Last Statement

C.Inside a parameterized contructor only

D.Can't be placed at any line in contructor

Ans :A-C

Packages:

56) Ria has a class called ‘Account.java’ under tow separate packages com.infy.debit and com.infy.credit.

Can she use the Account class of both the packages in another class called ‘ReportUti.java’ of package com.infy.util?

Option A; Yes, she can use

Option B: No, she cannot as there will be a compliation eror stating the import collides with another import

Option C: No, she cannot. The code will pass compilation but an ambiguity will get encountered during the execution.

Option D: No, she cannot as there will be a compilation error whiel creating Account class for the second time through in a different

Ans: Option B

Super:

72. Which of the following Java component can't be referred using ‘super’ keyword?

A.public contructor

- B.protected method
- C.private method
- D.protected instance variable

Ans : C

Lambda:

76. Which of the following is a valid lambda expression?

- A.(sum) → true
- B.x,y → true
- C.sum → {return 1 ==1}
- D.(a,b) → {int result; return result>0;}

A,C,D

Exception Handling:

73. Which of the following are the advantage of exception handling in Java(choose any 3 option)?

- A.To maintain the normal flow of execution
- B.Meaningful error reporting
- C.To document compile time errors
- D.To prevent the abrupt termination of a program

Ans : A,B,D

Java:

1. What will be the output of the below code?
//Assume all the required import statements are added
Public class TestClass
@Test
Public void test(){
String a = “”;
AssertSame(“a”,a);

Ans: An exception is thrown at runtime as assertSame() is not a valid function

2. What will be the output of the below code?

```
Public class Demo{  
    Public static void main(String args[]){  
        Int i = 34;  
        Int j = 34;  
        System.out.println(i==j);  
        Integer i1 = new Integer(34);  
        Interger i2 = new Integer(34);  
        System.out.println(i1==i2);  
  
    }  
}
```

Ans: true false

3. Which out of the following are true in regard to interfaces in Java?

- (i) An interface can contain default and static method with defined bodies.
- (ii) An object can be created of an interface
- (iii) Multiple inheritance is allowed in Interfaces
- (iv) Multi-level inheritance is not possible in interfaces

Ans: (i),(iii) & (iv)

4. Given

```
Public class Sample{  
    Public static void main(String[] args) throws Exception{  
        Try{  
            System.out.println("In try block");  
            System.exit(0);  
        }catch(Exception ex){  
            System.out.println("In catch block");  
            Ex.printStackTrace();  
        }finally{  
            System.out.println("In finally block");  
        }  
    }  
}
```

Predict the output?

Ans: In try block

5. Which of the following are the advantages of exception handling in Java(Choose any 3 options)?

- (i) To maintain the normal flow of execution
- (ii) Meaningful error reporting
- (iii) To document compile time error
- (iv) To prevent the abrupt termination of a program

Ans: (i),(ii) &(iv)

6. Predict the output of the below code:

```
final class Square{
    private double length,breadth;
    public Square(double length,double breadth){
        this.length = length;
        this.breadth = breadth;
    }
    Square(Square square)
    {
        System.out.println("Copy constructor invoke");
        length = square.length;
        breadth = square.breadth;
    }
    public String toString(){
        return "("+length+"x"+breadth+"i)";
    }
}
class Main{
    public static void main(String[] args){
        Square square1 = new Square(110,115);
        Square square2 = new Square(square1);
        System.out.println(square2);
    }
}
```

Ans: Copy constructor invoked

(110.0 + 115.0i)

7. Which of the following is a valid lambda expression?

Ans: (a,b) -> {int result; return result>0;}

8. What is the output of the following code?

```
Class Employee{
    Void display(char c){
        System.out.println("Employee name starts with: " +c+ ".");
        System.out.println("His experience is : 11 years.");
    }
}
Class Main extends Employee{
    Void display(char c){
        Super.display();
    }
}
```

```

        System.out.println("Another employee name also starts with: " +c+".");
        New Employee().display('D');
            Display(7);
    }
    String display(int c){
        System.out.println("His experience is: "+c+" years.");
        Return "Bye";
    }
}
Public class Demo{
    Public static void main(String a[]){
        Employee emp = new Main();
        Emp.display('S');
    }
}
Ans: Employee name starts with : S.
His experience is : 11 years.
Another employee name also starts with : S. Employee name starts with : D.
His Experience is : 11 years.
His Experience is : 7 years.

```

9. What is the output of the below code?

```

Import java.util.HashSet;
Public class TestDemo{
    Public static void main(String...sss){
        HashSet myMap = new HashSet();
        String s1 = new String("das");
        String s2 = new String("das");
        NameBean s3 = new NameBean("abcdef");

```

```

        NameBean s4 = new NameBean(:abcdef");
        myMap.add(s1);
        myMap.add(s2);
        myMap.add(s3);
        myMap.add(s4);
        System.out.println(myMap);
    }
}

Class NameBean{
    Private String str;
    NameBean(String str){
        This.str = str;
    }
    Public String toString(){
        Return str;
    }
}

```

Ans: [abcdef,das,abcdef]

10. Which among the following is/are true about Design Pattern

- i. Design pattern depends upon abstraction
- ii. Design patterns are completed designs that can be transformed directly into code
- iii. Design pattern depends on abstraction, follows the process of Dependency Injection
- iv. Design pattern is a template of solving problem that can be used in many real world software development

Ans i & ii

11. From the below options identify the methods and constructors in Throwable that support chained exceptions

- i. Throwable.getCause()
- ii. Throwable.initCause(Throwable)
- iii. Throwable(String, Throwable)
- iv. Throwable(Throwable)

Ans : Options i, iii & iv

12. What is Magic Number in Java in the context of java programming best practice

- i. A number which gets printed on the console
- ii. A direct usage of the number in the code
- iii. A number which magically disappears from the code
- iv. A number which is generated through error

Ans: Option2

A direct usage of the number in the code

13. Default format of LocalDate is

- I. yyyy,mm,dd
- II. yyyy.mm.dd**
- III. yyyy.dd.mm
- IV. yyyy,dd,mm

Ans Option ii

14. Which among the following option are correct with respect to HashMap

- I. Override boolean equals(Object o)
- II. Override toString()
- III. Override int hashCode()
- IV. Override String hashCode()

Ans i, Override boolean equals(Object o)

15. Which of the foll interfaces are not a part of java collections framework(choose any2)

- i. List
- ii. Queue
- iii. SoretdList
- iv. ArrayList

Ans: Option iii & iv

16. Where are string objects stored in memory using “new” keyword?

- i. In Stack Memory
- ii. In String constant pool in Heap Memory
- iii. In Native Methods stack Memory
- iv. Anywhere in Heap Memory**

Ans iv

17. What will be written at the line1 so that the below code will compile and run successfully?

```
Public class Main{  
//line1  
static {  
X[0] = 102;
```

```
    }
    public static void main(String [] args){
        System.out.println(x[0]);
    }
}
```

Ans : option ii, static int[]x = new int[3];

18. Given

```
public class Employee{
    private String roles;
    Public Employee (String role) {this.role= role;}
    Public Boolean equals (Employee emp){return emp.role.equals(this.role)}
}
}
```

Which among the following are correct with respect to the above code(Choose 2)

- I. Code give compilation error due to private attribute emp.role cannot be accessed
- II. Object.equals() method is not properly overriden
- III. Code will compile successfully
- IV. hashCode() method implementation is required

Ans iii & iv

19. Output of the below code

```
Public class TestDemo{
    Public static void main(String [] args){
        Integer n1 = new Integer (100);
        Integer n2= new Integer (100);
        Integer n3 =127;
        Integer n4 =127;
        System.out.println(n1==n2)
        System.out.println(n3==n4)
    }
}
```

Ans: false, true (option1)

20. Given

```
Public class ExceptionDemo{
    Static class Car implements AutoCloseable{
```

```

        Public void close(){
            System.out.println("Car door close")
            Throw new RuntimeException();
        }
    }

    Public static void main (String [] args){
        Try{
            //Line1

        }catch (Exception e){

            System.out.println("catch exception");
        }finally{

            System.out.println("finally")
        }
    }

    I.   Car car = new Car();
        System.out.println("try block")

    II.  Car car = new Car();
        Car.close();
        System.out.println("try block")

    III. Car car = new Car();

    IV.  System.out.println("try block")

```

Ans: all four

21. Given an abstract class Customer as below

```

Public abstract class Customer{
    Public abstract String getCustomerType();
}

```

Select a valid implementation of getCustomerType method in another class from the below options

- a. Abstract class C1 extends Customer{
 Public abstract String getCustomerType(){ return "Premium";}
 }
- b. Customer customer = new Customer(){
 Public String getCustomerType(){ return "Premium";}
 }

- c. class C1 extends Customer{

 Public abstract String getCustomerType(){ return "Premium";}

}

- d. new Customer(){

 return "Premium";}

Ans: a & c

22. Given below

```
Public class Demo{
@BeforeClass
Public static void afterClass(){
System. Out.println(1)
}

@Before
Public void before(){
System. Out.println(3)
}

@Test
Public void test(){
System. Out.println(5)
}

@After
Public void fter(){
System. Out.println(4)
}
@AfterClass
Public static void afterClass(){
System. Out.println(2)
}
```

Ans Option 1 :

1
4
5
3
2

23. Which of the following is 'FALSE' regarding 'super' keyword in Java?

Option3, Super keyword can be used to call a parent class protected constructor which is present in the same package

24. Which of the foll data structure is used by varargs in java

Ans: Option3, ArrayList

25. Which of the following is 'FALSE' regarding 'this' keyword in Java?

Ans option4, 'this' keyword can be used to create a block of code.

26. How many number of values can be accommodated by the varargs in java

Option4, Any number of values

27. Which of the foll statement is FALSE(Choose 2 options)

- I. An interface can extend from only one interfaces
- II. A class can extend from another class and at the same time implement any number of interfaces
- III. A class can extend multiple abstract classes
- IV. Many classes can implement the same interface

Ans i and iii

28. Have a look at the foll code and choose the correct option

```
Class ReportUtil{  
    Int reportId =0;  
    Static {++reportId}  
    Static int employeeReportId;  
    Public void preReport(){  
        employeeReportId= reportId;  
    }  
}
```

Ans: option 3 The code will not get compiled as instance variable cannot be referred from the static block

29. Which of the below methods can be defined in the single class in java. Select most suitable(select 2 options)

- 1. Void add(int,int)
- 2. Void add(int,int, int)
- 3. int add(int,int)
 - I. 1 & 3 together
 - II. 1 & 2 together
 - III. 2 &3 together
 - IV. 1,2 &3 together

Ans Options ii and iii

30. Ria has a class called Account.java under two separate packages com.infy.debit and com.infy.credit. Can she use the Account class of both the packages in another class called ReportUtil.java of package com.infy.util

Option2: No, She cannot as there will be a compilation error stating that the import collides with another import

31. Which of the following are true about enums in Java? (Choose 3 correct)

- I. enums can implement any other interface in java
- II. an instance of enum can be created outside of enum itself
- III. enum cant extend any other class except the abstract base class java.lang.enum
- IV. enum can be used to implement singleton design pattern

Ans i, iii & iv

32. What's is true with respect to abstract class being given below

```
Abstract class Employee{  
    //fields and constructor  
    Public void salaryComputer()  
    {  
        // code goes here  
    }  
    Public abstract void taxReduce()  
    {  
        // code goes here  
    }  
    Public abstract void benifitsInclude();  
}
```

Ans: option4. Abstract method of class Employee has definition

33. Which of the following are NOT good practice for creating objects.

- a. Lazy initialization of objects
- b. Creating String literals instead of String objects
- c. **Creating Wrapper objects instead of primitives**
- d. invoking static factory methods for immutable classes

Ans option c

34. Which among the foll are valid lamda expressions to sort the numbers in numberlistin descending order choose 3

- a. numberList.sort((x,y)->x.compareTo(y))
- b. numberList.sort(int x,int y)->x.compareTo(y))
- c. numberList.sort((x,y)->{return x.compareTo(y)})
- d. numberList.sort((Integer x, Integer y)->x.compareTo(y))

Ans

35. What is the output for the below code

```
public class VarArgsDemo{  
  
    static void func(int... x)  
    {
```

```

System.out.println("Number of arguments "+x.length);

for(int i:x)
    System.out.print(i+"");
    System.out.println();
}
void func(int a)           //Line1
{   System.out.println("one"); }

public static void main(String[]args){
    new VarArgsDemo().func(150);
    func(11, 12, 13, 14);
    func();
}

```

Ans:Option B

```

Number of arguments1
150
Number of arguments4
11121314
Number of arguments0

```

36.Given, Assume all the required imports are added

```

Public class TestDemo{
    Private AaaryList list //Line1
    @Test //lin2
    Public void test(){
        assertTrue(list.isEmpty());
    }
}

```

Which of the following is true regarding the above code fragment

Ans:if we replace Line2 by “@Test(expected = NullPointerException.class)” the test case will pass

37. Given

```

HashMap <Integer, Integer> myMap = new HashMap<Integer, Integer>();
myMap.put(1001,5);
myMap.put(1002,8);
myMap.put(1002,5);
myMap.replace(1002,5, 100);
    system.out.println(myMap)

```

Ans: {1001=5, 1002=100}

38. What is the result when the foll code is compiled?

```

Class Student extends Exception{}
Class Hosteller extends Student{}
Public class StudentTester{
    Public static void main (String args[])
}

```

```

Try{
    //some monitored code
    Throw new Hosteller();

}Catch(Student st){
    System .out.println("Student class Exception");
}catch (Hosteller host){
    System .out.println("Hosteller class Exception");
}
}
}

```

Ans: The code will not compile Unreachable catch for Hosteller because Student class exception is caught before Hosteller

39.

```

class Operations{
    Public void addition(){}
}

```

Class AdvOperations extends Operations{
Void addition()//line1

```
{
}
```

Line 1 generates compilation error . which of then below option helps to resolve this

Ans: public keyword has to be added

40. What will be in line1 to get the output as 150. Choose 2

```

class EmployeeUtil {
    //line1
    double amountTax;
    public double taxCalculator(double salary)
    {
        amountTax = salary *TAX_PERCENTAGE/100;
        return amountTax;
    }
}
public class ConstantMain
{
    public static void main(String[] args) {
        System.out.println(new EmployeeUtil().taxCalculator(5000));
    }
}

```

Ans:

```
final int TAX_PERCENTAGE=3;  
static int TAX_PERCENTAGE=3;
```

41.

```
interface Component {  
    String cname = "Motor";  
    String getName(String name);  
}  
  
public class Demo implements Component{  
    public String getName(String name) {  
        System.out.println("Inside Demo Class");  
        return "Componenet from interface is :" + cname+"and component  
from class is "+ name+ ";"  
    }  
  
    public static void main(String[] args) {  
  
        Demo demo = new Demo();  
        System.out.println(demo.getName("Battery"));  
        demo.getName("Battery");  
  
    }  
}
```

Ans Inside Demo Class
Componenet from interface is :Motorand component from class is Battery;
Inside Demo Class

42.

```
public class CollectionTest {  
  
    public static void main(String[] args) {  
  
        Collection<Integer> employeeCollection = new HashSet();  
        collection.add(1001);  
        collection.add(1002);  
        collection.add(1001);  
  
        Set<Integer> newSet = new TreeSet();  
        newSet.addAll(employeeCollection);  
        System.out.println(newSet);  
  
    }  
}
```

Ans : Runtime Exception : null pointer Exception

43. Which of the following regarding an abstract class is/are true in Java

i) Object of an abstract class cant be created

ii) An abstract class is designed only to act as a base class in hierarchy to be inherited by other classes

Ans : Both i & ii

44.

```
public class CodeForException {
    //public static void main(String[] args) {
    public void callMe() throws Exception{
        try
        {
            int value=3/0;
        }
        catch (ArithmaticException aa)
        {

            System.out.println(aa);
        }
    }
    public void calling()
    {

        callMe();
    }
}
```

Ans : Compilation error

45.

```
public class Demo {
    private static String id;
    private Random random = new Random();

    public Demo(){

        if(id ==null){ }

        id = "ACC1101"+ Math.abs(random.nextInt());
    }
}

public String getid() {
    return id;
}
```

```
Ans - if(id==null){
```

46. Given

//Assume all the required imports are added

```
public class TestDemo {  
    private ArrayList list;//Line1  
    @Test //Line2 (expected = NullPointerException.class)  
    public void test(){  
        assertTrue(list.isEmpty());  
    }  
}
```

Which of the following statement is true regarding the above code fragment?

```
Ans: if we replace Line2 by "@Test(expected = NullPointerException.class)" the  
test case will pass
```

47.

```
public static void main(String args[]){  
    HashMap<Integer, Integer> myMap = new HashMap<Integer, Integer>();  
    myMap.put(1001, 5);  
    myMap.put(1002, 8);  
    myMap.put(1002, 5);  
    myMap.replace(1002, 5, 100);  
    System.out.println(myMap);  
}
```

```
Ans: {1001=5, 1002=100}
```

48. what is the result when the following code snippet is compiled?

```
class Student extends Exception{}  
class Hosteller extends Student{}  
public class StudentTester {  
    public static void main(String[] args){  
        try{  
            //some monitored code  
            throw new Hosteller();  
        }  
        catch (Student st){  
            System.out.println("Student class exception");  
        }  
        catch (Hosteller host){  
            System.out.println("Hosteller class exception");  
        }  
    }  
}
```

Ans: The code will not compile Unreachable catch for Hosteller because Student class exception is caught before Hosteller

49.

```
class Operations
{
    public void addition()
    {}
}
class AdvOperations extends Operations {
    void addition()//Line1
    {}
}
```

Line 1 generates Compilation error. Which of the below options helps to resolve this?

Ans: public keyword has to be added

50.

```
Public class TestDemo{
    Public void main(int x){
        System.out.println("Main1")
    }
    Public static void main(String args[]){
        System.out.println("Hello Main")
    }
}
```

Ans : option 4 : Hello Main

51.

```
import java.util.HashMap;
import java.util.Iterator;
import java.util.TreeSet;
public class ABC {
    public static void main(String args[])
    {
        TreeSet set = new TreeSet();
```

```

        set.add("a");
        set.add(6);
        set.add("c");
        Iterator ite =(Iterator) set.iterator();
        while(ite.hasNext()){
            System.out.println(ite.next());
        }
    }
}

```

Ans: The code will give `java.lang.ClassCastException: java.lang.String incompatible with java.lang.Integer`

52. what is the output of the following code?

```

public class Demo5 {
    public static void main(String[] args){

        int x[][] = new int [4][]; //Line 1
        x[0] = new int [1];
        x[1] = new int [2];
        x[2] = new int [3];
        x[3] = new int [4];
        int a,b,c=0;
        for(a=0; a<4; a++){
            for(b=0; b<a+1; b++){
                x[a][b] = c;
                c++;
            }
        }
        for(a=0; a<4; a++){
            for(b=0; b<a+1; b++){
                System.out.print(" " + x[a][b]);
                c++;
            }
        }
        System.out.println();
    }
}

```

Ans:

0
12
345
6789

53. which of the following keyword is used to prevent the content of a variable from being modified from outside.

Ans - Final

54.

```
class Book{  
    int bookid =2356;  
  
}  
class Book1 extends Book{  
    int bookid = 1167;  
}  
class Book2 extends Book1{  
    int bookid = 2378; //Line8  
    void display(){  
        System.out.println(super.super.bookid); //Line10  
        System.out.println(super.bookid); //Line11  
        System.out.println(bookid);  
    }  
}  
public class Demo2 {  
    public static void main(String[] args){  
        Book2 book2 = new Book2();  
        book2.display();  
    }  
}
```

Ans - Compilation fails, because of an error in Line10 as super keyword is unexpected.

55. Consider the below class and identify the extension of the output file when we execute the command javac Employee.java

```
class Employee1 {  
    private int x=10;  
    public void showX(){  
        System.out.println(x);  
    }  
}
```

Ans - .java

56. which of the below if statement is used to find a year is a leap year or not

Ans: if((y%4==0) && (y%100 !=0) || (y%400==0))

57. Analyze the below code and predict the output when executed the code

```
public class Demo6 extends Book{  
    int bookid = 4362;  
    public int getvarchar(){  
        return bookid;  
    }  
}
```

```

public void Cell(){
    System.out.println(getvarchar());//Line1
}
public static void main(String[] args){
    Book book = new Book();
    super.Cell(); //Line 2
}
class Book{
    int bookid = 17252;
    public int getvarchar(){
        return bookid;
    }
}

```

Ans: Compilation error in line 2 as super keyword cannot be used in static context

58. have a look at the following class and predict what should be placed at line 1 to get 150.0 as output when the code gets executed? (choose 2 options)

```

class EmployeeUtil{
    //line1
    double amountTax;
    public double taxCalulate(double salary){
        amountTax = salary*TAX_PERCENTAGE/100;
        return amountTax;
    }
}
public class ConstantMain{
    public static void main(String[] args){
        System.out.println(new EmployeeUtil().taxCalulate(5000));
    }
}

```

Ans:

```

static int TAX_PERCENTAGE=3;
final int TAX_PERCENTAGE=3;

```

59. if the child class of an abstract class does not implement all its abstract methods then it should be declared as

Ans: Option A: Abstract class

60. Which of the following statements regarding an abstract class is/are true in Java

- I. Object of an abstract class cant be created
- II. An abstract class is designed only to act as a base class in hierarchy to be inherited by other classes

Ans : Both I & II

61. which of the following is valid function used to read values using Scanner in java (choose 3)

- 1. **nextInt()**
- 2. **nextChar()**
- 3. **nextLong()**
- 4. **nextLine()**

Ans: option 1,3 and 4

62. What will be output of the below code snippet?

```
package Example;

public class Pet {
    public void displayName(){
        System.out.println("Inside Pet");
    }
}

package java.pac;
import Example.Pet;
public class Dog extends Pet{
    @Override
    public void displayName(){
        System.out.println("Inside Dog");
    }
}

package com.infy;
import Example.Pet;
public class Demo7 {
    public static void main(String[] args){
        Pet pet = new Dog(); //Line1
        pet.displayName();
    }
}
```

```
 }  
  
}
```

Ans – Compilation error at Line1 due to ClassCastException

63. Which of the foll is FALSE regarding polymorphism in JAVA

- a. Polymorphism is the ability of an object to take different forms
- b. Polymorphism can be as a single action that can be performed in different ways
- c. Polymorphism means different forms by creating different classes that are referred to each other by inheritance
- d. Polymorphism is using same method multiple times in a class with different parameters

Ans Option D

64. what is the result of the following code?

```
public class Vehicle {  
    static class Car{  
        public void go(){  
            System.out.println("Car Ignition");  
        }  
    }  
    static class ElectricCar extends Car{  
        public void go(){  
            System.out.println("ElectricCar Ignition");  
        }  
    }  
    static class PetrolCar extends Car{  
        public void go(){  
            System.out.println("PetrolCar Ignition");  
        }  
    }  
    public static void main(String[] args){  
        Car car = new ElectricCar();  
        car.go();  
    }  
}
```

Ans- ElectricCar Ignition

65. If child class of abstract class doesnot implement all its abstract methods then it should be declared as?

Ans – abstract class

66. which are valid upper bound by class Employee of list. (Doubt)

Ans - List<?extends Employee>

Choose any 2

```
public class App {
    public static void main(String[] args) {
        String msg = null;
        try {
            System.out.println(msg.length());

        }catch(NullPointerException ex) {
            System.out.println("Exception is caught here");

            throw ex; //Line 1
            System.out.println(msg); //Line 2
        }
    }
}
```

Ans:

//a, b

A.Place line2 before line 1

B.Remove line 2 ie after throw, there should not be any statements

```
interface Book {
    static void bookName() {
        System.out.println("in interface book");
    }
}
```

```
public class BookImpl implements Book{
    void bookName() {
        System.out.println("In BookImpl Class");
    }
}
```

```
public class BookApp {
```

```
public static void main(String[] args) {
    // TODO Auto-generated method stub
    new BookImpl().bookName();
}

}
```

ANS: code compiles and prints “In BookImpl Class”

Which of the below is not a valid classification of design pattern

Creational Pattern

Structural pattern

Behavioral Pattern

Java pattern

ANS: D Java pattern

Section 1:

- What is the output of the following code?

```
public class Main{
    public static void main(String args[]){
        int twoD[][] = new int [4][];
        //Line 1
        twoD[0] = new int [1];
        twoD[1] = new int [2];
        twoD[2] = new int [3];
        twoD[3] = new int [4];
        for (int i=0;i<4;i++){
            for (int j=0;j<i+1;j++){
                twoD[i][j]; //Line 2
            }
        }
    }
}
```

```

        }
        System.out.println("executed");
    }
}

```

Ans – Compilation error in Line 2 as there is no Assignment operator used

2. Which of the following statement is false about an object in java?

Ans – Object can communicate with each other

3. What is the output of the code given below?

```

public class ABC {
    public static void main(String args[])
    {
        boolean flag = false;
        if(flag = true)
        {
            System.out.println("true");
        }
        else
        {
            System.out.println("false");
        }
    }
}

```

Ans - true

4. Ria has a class called ‘Account.java’ under two separate packages. Com.infy.debit and com.infy.credit can see give the account.java of both the packages in another class called ‘ReportUtil.java’ of package com.infy.util?

Ans – No She cannot as there will be compilation error stating the import collides with another import.

5. Identify the suitable datatype to be used for variable ‘b’ at line 1?

```

class Expression {
    public void calc()
    {
        double x=10;
        int y = 20;
        float z= 30;
        //Line 1 --- Answer is double
        b=x+y+z;
    }
}

```

Ans – double

6. What is the result when the following code is compiled and executed?

```
class Demo {  
    int x=1;  
    int y=2;  
    Demo display(Demo demoParam){  
        Demo obj = new Demo();  
        obj = demoParam;  
        obj.x = demoParam.x++ + ++ demoParam.y ;  
        demoParam.y = demoParam.y;  
        return obj;  
    }  
    public static void main(String[] args){  
        Demo obj1 = new Demo();  
        Demo obj2 = obj1.display(obj1);  
        System.out.println("obj1.x = " + obj1.x + "obj1.y = " + obj1.y);  
        System.out.println("obj2.x = " + obj2.x + "obj1.y = " + obj2.y);  
    }  
}
```

Ans:

```
obj1.x = 4obj1.y = 3  
obj2.x = 4obj1.y = 3
```

7. Have a look at the following class and predict the option that is correct?

```
public class CodeForException {  
    public void callMe() throws Exception{  
        try{  
            int value = 3/0;  
  
        }catch (ArithmeticException ae){  
            System.out.println(ae);  
        }  
    }  
    public void calling(){  
        callMe();  
    }  
}
```

```
    }  
}  
}
```

Ans: The code will face issues during compilation as the calling code neither handles nor throws Exception

8. What will be output of the code when executed?

```
public class OperatorTester {  
    public static void main(String[] args){  
        int a = 10;  
        int b = 15;  
        if (++a <(b=b-=4) || (a=a+=4)>b++){  
            System.out.println(a+", "+b);  
        }  
    }  
}
```

Ans – 15,12

9. What is the result when the following code snippet is executed?

```
class Employee {  
    static String name = " ";  
    public static void main(String[] args) throws Exception{  
        try{  
            name+="jo";  
            throw new Exception();  
  
        }catch(Exception e){  
            name+="hn";  
  
        }finally{  
            name+="s";  
            empName();  
            name+="on";  
        }  
        System.out.println(name);  
    }  
    static void empName(){  
        throw new NullPointerException();  
    }  
}
```

Ans – The code will give java.lang.NullPointerException in finally block when executing

10. What is the result when the code is compiled and executed?

```
class Calculator {  
    int a = 123, b=200;  
    public void display(){  
        System.out.println("a:" + a + "b:" + b+"");  
    }  
}  
class CalculatorDemo{  
    public static void main(String[] args)  
    {  
        Calculator calculator1 = new Calculator(); //Line1  
        Calculator calculator2 = calculator1; //Line2  
        calculator1.a+= 1;  
        calculator1.b+= 1;  
        System.out.println("calculator1 values:");  
        calculator1.display();  
        System.out.println("calculator2 values:");  
        calculator1.display();  
    }  
}
```

Ans –
calculator1 values:
a:124b:201
calculator2 values:
a:124b:201

Section2

1. What can be expected when the following code is compiled

```
abstract class Customer {  
  
    public int custId;//Line1  
    Customer()//Line2  
    {  
        custId=23456;//Line3  
    }  
    abstract public void setId();  
    abstract final public void getId();//Line4  
}
```

Ans : Option D: Compilation error in Line 4 abstract methods cant be final

2. What is the correct way of placing 'this' keyword in a constructor?

Ans Option A : First Statement

3. At what position should Varargs be placed in a parameterised method

Ans Option B: Last Place

4. Predict the output of the following code

```
public class Demo {  
    static int x=232;  
    int y=135;  
    public void display(){  
        System.out.print("Inside Demo");}  
    public static void show(){  
        System.out.print(x);  
    }  
  
    public static void main(String[] args)  
    {  
        Demo.show(); //Line 1  
        Demo demo=new Demo();  
        demo.show(); //Line2  
        show(); //Line3  
        demo.display();  
    }  
}
```

Ans: Option A: Compilation error in line2 as static method cannot be called with object reference

5. Which of the following keyword can be used to restrict a class to be implemented in java

Ans Option B. Final

6. Given below. What will be the output when code is compiled and executed

```
public class Parent {  
  
    public final void show(){  
        System.out.println("show() inside Employee");  
    }  
}  
  
final class Child extends Parent {  
    public void show1() { //Line1  
        final int x=100;  
        System.out.println("show() inside Unit");  
        System.out.println(x);  
    }  
}
```

```

    }}
public class Demo {

    public static void main(String[] args) {
        Parent parent = new Child();
        new Child().show1(); //Line2

    }}

```

Ans: Option D

show() inside Unit

100

7. Which of the following is FALSE regarding parameterized constructor in Java

- a. parameterised constructor should have void as return type
- b. parameterised constructor can take any number of parameters
- c. parameterised constructor cannot have private access modifier
- d. parameterised constructor throw an exception

Ans Option c parameterised constructor cannot have private access modifier

8. What is the output of the following

```

public class Trainer{

    public void display(String name) {
        System.out.println("I am a trainer");
        print(name);
    }

    public void print(String name) {
        System.out.println(" I train "+ name+ ".");
    }
}

public class Trainee extends Trainer {
    String myname;

    public Trainee(String myname) {
        super();
        this.myname= myname;
    }

    public void display(String name) {
        super.display(name);
        System.out.println("I am a trainee");
    }
}

```

```

        print("Java");
    }

public void print(String name) {
    super.print(name);
    System.out.println("I want to learn " +name+"");
}

public static void main(String[] args) {
    Trainer trainee = new Trainee("XYZ");
    trainee.display("Java");

}

```

Ans: Option B

```

I am a trainer
I train Java.
I want to learn Java
I am a trainee
I train Java.
I want to learn Java

```

9. Which of the following is FALSE regarding ‘this’ keyword in java?

Ans--- optionD, ‘this’ keyword can be used to create a block of code.

10. Given

```

Class Parent{}
Class Child extends Parent{}
Final class GrandChild extends Child{}

```

Which of the following statement is not ‘true’ about the above code.

Ans : Option C: Reference of parent class can accept the instance of child class but not the instance of GrandChild class

Section 3

1. What will be the output of the below code

```

public class student {

```

```

String stuName="Jacklin";

void func() throws Exception{
    try
    {
        stuName+="--";
    }
    catch (Exception e) {
        throw new Exception();
    }
    finally
    {
        stuName+="Hello" +stuName;
    }
    stuName+="!!!!";
}

void disp() throws Exception
{
    func();
    System.out.println(stuName);
}

public static void main(String[] args) {
    try
    {
        student student = new student();
        student.disp();
    }
    catch (Exception e) {
        System.out.println("CatchBlock");
    }}}

```

Ans Jacklin--HelloJacklin--!!!!
Option D

2. Predict the output of the foll

```

public class Greeter{

    public static void main(String[] args) throws Exception{
        try
        {
            System.out.println("Greeting"+ ""+args[0]);
        }catch (ArrayIndexOutOfBoundsException e) {

            System.out.println("Sam");
        }
    }
}

```

Ans : Option C Sam

3. Predict the output of the below code.

```
public class TestDemo{  
  
    public static void main(String[] args) {  
        for(int a=0;a<6;++a)  
            try {  
                if(a%3==0)  
                    throw new Exception("Except1");  
                try {  
                    if(a%3==1)  
                        throw new Exception("Except2");  
                    System.out.println(a);  
  
                }catch (Exception inside) {  
                    a +=2;  
                }finally {  
                    ++a;  
                }  
            }catch (Exception outside) {  
                a+=3;  
            }finally {  
                ++a;  
            }  
    }  
}
```

Ans: Option A: 5

4. Given

```
Public class Sample{
```

```
    Public static void main(String[] args) throws Exception{  
        Try{  
            System.out.println("In try block");  
            System.exit(0);  
        }catch(Exception ex){  
            System.out.println("In catch block");  
            Ex.printStackTrace();  
        }finally{  
            System.out.println("In finally block");  
        }  
    }
```

```
    }
}
Predict the output?
Ans: Option B.In try block
```

5. Which of the following exceptions are ignored during compile time (choose any 2 option)

Option A and C
ArrayIndexOutOfBoundsException
NullPointerException

Section 4

1. Identify the outcome of the given code snippet

```
public class Demo{

public static void main(String[] args) {
int [] arrVar = {11,22,33,44,55,66,77,88,99,109};
int position =3;
int value =7;
for(int i= arrVar.length-1;i>position;i--) {
    arrVar[i]= arrVar[i-1];
}
arrVar[position]= value;
System.out.println("New Array"+ Arrays.toString(arrVar));
}
}
```

Ans :
Option D New Array[11, 22, 33, 7, 44, 55, 66, 77, 88, 99]

2.Which of the following are true about enums in Java? (Choose 3 correct)

- I. enums can implement any other interface in java
- II. an instance of enum can be created outside of enum itself
- III. enum cant extend any other class except the abstract base class java.lang.enum
- IV. enum can be used to implement singleton design pattern

Ans Options i, iii & iv

3.Predict the output of the below

```
public static void main(String[] args) {
String name1 ="Java";
String name2 = "Java";
System.out.println(name1==name2);
```

```
        System.out.println(name1.equals(name2));
    }
```

Ans Option A: True True

4. Given

```
public class App{

public static void main(String[] args) {
    String s1 = new String ("smart");
    String s2 =s1;
    if(s1==s2) {
        System.out.println("==smart");
    }
    if (s1.equals(s2)) {
        System.out.println("equals smart");
    }
}
```

Ans Option B

```
==smart
equals smart
```

Section 5

1.What will happen when the following code is executed?(Output – not sure)

```
public class TestDemo{

public static void main(String[] args){
    List list1=new ArrayList<>();
    list1.add("1");
    list1.add("2");
    list1.add(1, "3"); //Line1
    List list2=new LinkedList<>(list1); //Line2
    list1.addAll(list2); //Line3
    list2=list1.subList(2,6); //Line4
    list2.clear();
    System.out.print(list1+ "");
```

```
    }  
}
```

Ans: [1, 3] (answer not in option)

2. Which among the following option are correct with respect to HashMap
Override boolean equals(Object o)
Override toString()
Override int hashCode()
Override String hashCode()

Ans – option a - Override boolean equals(Object o)

3. Identify the incorrect statement as per the collection Framework hierarchy?

Ans D: LinkedHashSet Implements HashSet

4. What is the result of compiling and running this code snippet?

```
import java.util.Arrays;  
import java.util.Comparator;  
  
public class Demo3 implements Comparator<String>{  
    @Override  
    public int compare(String x, String y){  
        return x.toLowerCase().compareTo(x.toLowerCase());  
    }  
    public static void main(String[] args){  
        String[] values = {"JOHN", "Annie", "JACKLINE"};  
        Arrays.sort(values,new Demo3());  
        for (String str:values)  
            System.out.print(str + " ");  
    }  
}
```

Ans - JOHN Annie JACKLINE

5. Predict the output of the below code snippet?

```
import java.util.ArrayList;  
import java.util.List;  
  
public class sam1 {  
    public static void main (String[] args){  
        List<String> list = new ArrayList();  
        list.add(0,"A");  
        list.add(1,"B");
```

```

        list.add(1,"C");
        for (Object object:list){
            System.out.print(" " + object);
        }
    }
}

```

Ans --- A C B

Section 6:

1.What is true regarding the following code snippet

```

public interface StaticInterface {

    static void staticMethod()
    {
        System.out.println("Inside interface");
    }
}

public class StaticInterfaceImpl implements StaticInterface {

    public void staticMethod()
    {
        System.out.println("Inside class");
    }
}

public class StaticDemo {

    public static void main(String[] args)
    {
        new StaticInterfaceImpl().staticMethod();
    }
}

```

Ans: Option D. code will print “inside class” on execution

2. Which out of the following are true in regard to interfaces in Java
- An interface can contain default and static method with defined bodies
 - An object can be created of an interface
 - Multiple inheritance is allowed in interface
 - Multi level inheritance is not possible in interface

Ans Option A and C

3. Default format of LocalDate is

Ans Option B yyyy.mm.dd

4. Given

```
interface Greeting{
    default void greet() {
        System.out.println("In Greet interface");
    }
}

class GreetingDef implements Greeting{
    public void greet() {
        System.out.println("In GreetingDef class");
    }
}

public class App {

    public static void main(String str[]){
        Greeting obj = new GreetingDef();
        obj.greet();
    }
}
```

Ans:Option B- In GreetingDef class

5. The below code will generate compilation error. Select the possible options to avoid it(Choose 2)

```
public interface Insurance{

    static void policy() {
        System.out.println("policy");
    }
}

public class InsuranceImpl implements Insurance{
    public void policyPayment() {
        policy();
    }
}

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

- ```
 new InsuranceImpl().policyPayment();
 }
}

a. Static method of an interface can only be accessed using interfaces name
b. Static method should always be public
c. Static method cannot be invoked inside the non static method
d. Policy() method of interface has to be accessed using interface name
```

Ans option A & D

## Section 7:

1. What is Magic Number in java in the context of java programming best practices?

Ans – A direct usage of number in the code.

2. Given

```
public class Employee {
 private int empId;
 private String empName;
 private String designation;
 private transient String dob;
}
```

Analyze the given code and identify the suitable comments from the below option.

Ans:

- (i) Fields in non-serializable classes should not be ‘transient’
- (ii) Make the Employee class as serializable

3. From the below options identify the methods and constructors in Throwable that support checked exception?

- (i) Throwable getCause()
- (ii) Throwable initCause(Throwable)
- (iii) Throwable (String, Throwable)
- (iv) Throwable (Throwable)

Ans - (i), (iii), (iv)

4. Identify the valid code needs to be inserted in line5, assume the code is running in

```
import java.util.Random;
```

```
public class Emp {
 private static String id;
 private Random random = new Random();
 public Emp(){
 //line5 if(id==null){

 id = "ACC1101" + Math.abs(random.nextInt());
 }
 }
 public String getId(){
 return id;
 }
}
```

Ans - if(id==null){

5. Select the valid code fragment according to Java coding standard?

- (i) 

```
public void draw(String s){
 if(s.equals("Square")){
 drawSquare();
 }
}
```
- (ii) 

```
public void draw(String s){
 if("Square".equals(s)){
 drawSquare();
 }
}
```

Ans: Both (i) and (ii) are valid.

## Section 8:

1. Which among the following comes under Creational Design pattern?

Ans: Singleton Design Pattern

2. Which of the statements are true about design patterns?
  - i. There are only three design patterns defined for Java
  - ii. We can use each design pattern only once per application
  - iii. Design patterns are conceptual reusable solutions

Ans: Statement i,ii,iii

3. What changes need to be made In the following code to make the singleton pattern correct?(choose 2)

```
public class Employee1 {
 public static Employee1 employeeInstance;
 private Employee1(){}
 public static Employee1 getEmployee()
 {
 if (employeeInstance==null){
 employeeInstance = new Employee1();
 }
 return employeeInstance;
 }
}
```

Ans:

Option A &D

- A. None the singleton pattern is properly implemented  
D. Change the access modifier of employeeinstance from public to private

## Section 9:

1. Given

```
//Assume all the required imports are added
import java.util.ArrayList;

import org.junit.Before;
import org.junit.Test;

public class Demo4 {
 static int a = 10;
 static ArrayList b = new ArrayList();
 @Before
 public void init(){
 a=15;
 b.add(a);
 }
}
```

```

@Test
public void test(){
 a=a+20;
 System.out.print(a);
 System.out.println(b);
}
@Test
public void test1(){
 a=a+30;
 System.out.print(a);
 System.out.println(b);
}

```

Predict the output?

Ans -  
35[15]  
45[15, 15]

2. Which of the following annotation must be used in a test class to run same test again and again

Ans: `@Test`

3. Predict the output for the below

//Assume all the required import statements are added

```

import org.junit.Before;
import org.junit.Test;

```

```
import junit.framework.Assert;
```

```

public class TestDemo1 {
 @Before
 public void beforeTest1(){
 System.out.println("in before test2");
 }
 @Before
 public void beforeTest2(){
 System.out.println("in before test1");
 }
 @Test
 public void test(){
 String a = "123";
 Assert.assertEquals("123",a);
 }
}

```

Ans:

Test Passed as they point the same object and prints the below in console  
in before test1  
in before test2

## ThJava Question

1. Which of the following is not pre defined annotation in Java?

@Deprecated

@Overriden

@SafeVarargs

@FunctionInterface

**Ans: Option B(@Overriden)**

2. public class TestString3{

    Public static void main(String[] args){

        //insert code here//Line3

        System.out.println(s)

    }

}

Which of the below code fragment when inserted independently at line3 generate output as 498

**Ans Option B**

StringBuffer s = new StringBuffer("123456789").s.delete(0,3).replace(1,3,"98").delete(3,8);

3. Predict output of this

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

 String Value1= "Hello";
 String Value2= new String ("Hello");
 System.out.println(Value1.equals(Value2)+" "+(Value1==Value2));
 String Value3= Value2.intern();
 System.out.println((Value1==Value3)+" "+(Value1.equals(Value3)));

}
```

**Ans truefalse  
Truetrue**

66. Which of the following statements are true if a duplicate element obj T is added to a HashSet?

- a) The element obj T is not added and add() method returns false
- b) The element obj T is added successfully
- c) An exception occurs during runtime
- d) An exception occurs during compile time

. which of the following is incorrect regarding interfaces in Java SE8

a.all the methods are public,abstract by default

b.all the variables are public by default

c.methods can have implementation

d.its possible to hold static method

a) a and b

b) b and c

c) a,b and c

d) a only

Which of the below are NOT good practices for creating objects?

- a) Lazy initialization of objects
- b) Creating String literals instead of String objects
- c) Creating Wrapper objects instead of primitives
- d) invoking static factory methods for immutable classes

Which of the below statement indicate the need to use the factory pattern?

- a) we have two classes that do the same thing
- b) we only want one instance of the object to exist
- c) we want to build a chain of objects
- d) we don't want the caller to depend on a specific implementations

4.What is the output when the below code is compiled and executed

```
Package exceptions;
Public class Demo
{
public static void main(String[] args) {

 try
 {
 return;

 }
 finally
 {
```

```

 System.out.println("Finally");
 }

}

```

**Ans : Option A:( Finally)**

What is the output of the below code snippet

```

enum Customer
{
private CUSTID
public CUSTNAME
protected ADDRESS

}

```

**Ans : D Compilation Fails**

**(Explanation: Enum cannot have any modifiers. They are public, static and final by default)**

What is the output of the below code

```

package javaBasics;
public class student {

 String stuName="Jackin";
 void display()
 {
 try
 {
 stuName+="John";
 func();
 }
 catch (Exception e)
 {
 stuName+="GoodName";
 }
 }

 void func() throws Exception{
 try
 {
 stuName+=".....";
 method();
 }
 catch (Exception e)
 {
 throw new Exception();
 }
 finally
 {

```

```

 stuName+="!!!!!";
 }
 stuName+="hello";
}
void method() throws Exception
{
 throw new Exception();
}

void disp()
{
 System.out.println(stuName);
}

public static void main(String[] args) {
 try
 {
 student student = new student();
 student.display();
 student.disp();
 }
 catch (Exception e) {
 System.out.println("CatchBlock");
 }
}
} Ans Option D (JackinJohn.....!!!!GoodName)

```

What will be the output of the following code

```

public class Test{
 public void method()
 {
 for(int i=0;i<3;i++) {
 System.out.println(i);
 }
 System.out.println(i);
 }
}

```

**Ans: C. Compilation Fails**

What is the result of attempting to compile and run this program

```

public class Customer{
 public static void main(String [] args)
 {
 Float f1= new Float(67.65f);
 Float f2= new Float(36.45f);
 if(f1>f2)
 {

```

```

 System.out.println("f1 is bigger than f2");
 }
} else
{
 System.out.println("f1 is not bigger than f2");
}

}
}

```

**Ans : Option A (f1 is bigger than f2)**

What is the result of the following code snippet when compiled

```

public class Employee {
 int employeeid;
 double getEmployeeid()
 {

 System.out.println("Employee Id");
 return employeeid;
 }
}

```

Ans A: The code will not be complied as there is no main method

Given Enum definition and java class

```

enum Day{
 SUNDAY(1), MONDAY(2), TUESDAY(3), WEDNESDAY(4), THURSDAY(5), FRIDAY(6)
,SATURDAY(7) ;
 private int value;
 private Day(int value)
 {
 this.value =value;
 }

 public int getValue()
 {
 return this.value;
 }
}

```

```
public class Employee {
```

```

public static void main(String[] args)
{
 for (Day day: Day.values()) {

 //Line1
 System.out.print(day.toString()+"-");

 //System.out.print(day.name()+"-");
 }
}

```

What should be [placed in line 1 to get the output as  
 SUNDAY-MONDAY-TUESDAY-WEDNESDAY-THURSDAY-FRIDAY-SATURDAY-  
 Choose one or more options

**Ans : A and C**

```

System.out.print(day.toString()+"-");
System.out.print(day.name()+"-");

```

What will be the output of the following code

```

public void method(){
 for(int i=0;i<3;i++){
 System.out.print(i);
 }
 System.out.print(i);
}

```

**Ans: C. Compilation fails**

which of the below exceptions are mostly thrown by JVM in a Java application?(Choose all that apply)  
 means runtime exception

- a) ClassCastException
- b) IllegalStateException
- c) NumberFormatException
- d) IllegalArgumentException
- e) ExceptionInitializerError

Checked Exceptions are Compile time exceptions

Unchecked Exceptions are runtime exceptions

//Check Tutorial

What's the output of the following code

```
public class Demo {
 void main(){
 System.out.println("JAVA");
 }
 static void main(String args){
 System.out.println("Spring");
 }
 public static void main(String[] args){
 System.out.println("Hibernate");
 }
 void main(Object[] args){
 System.out.println("Apache Camel");
 }
}
```

**Ans Option A: Hibernate**

Given the below code snippet, predict the correct option

```
public class Operator {
 public static void main(String[] args){
 float val1=5.3f;
 float val2=2.3f;
 float result= val1 %val2;
 System.out.println(result);
 }
}
```

**Ans: Option A** **Code compiles, runs and produces the output 0.7000003**

What is the output for the below code

```
public class Employee {

 public final void show(){
 System.out.println("show() inside Employee");
 }
}
```

```

public class Dem011 {

 public static void main(String[] args) {
 Employee employee = new unit();
 new unit().show1();

 }
}

final class unit extends Employee {
 public void show1() {
 final int x=100;
 System.out.println("show() inside Unit");
 System.out.println(x);
 }
}

```

**Ans:Option D.**  
**show() inside Unit**  
**100**

What is the output when we execute the below code

```

public class Dem011 {
 static class Customer {
 public void go() {
 System.out.println("Inside Customer");
 }
 }
 static class Account extends Customer {
 public void go() {
 System.out.println("Inside Account");
 }
 }
 static class Branch extends Customer {
 @Override public void go() {
 System.out.println("Inside Branch");
 }
 }
 public static void main(String[] args) {
 Customer customer = new Account();
 Branch branch = (Branch) customer;
 branch.go();

 }
}

```

**Ans Option5: An exception is thrown at runtime because (Branch)Customer is incorrect**

Predict the output for the below code

```
public class Game {

 public static void main(String[] args) {
 displayRegistration("Hockey"); //Line 1
 displayRegistration("Kho-Kho", 132, 102, 36); //Line 2
 }

 public static void displayRegistration (String gameName, int... id) {
 System.out.println("Registration for " + gameName + ".");
 for(int i=0; i<id.length; i++) {
 System.out.println(id[i] + "");
 }
 }
}
```

**Ans**

```
Registration for Hockey.
Registration for Kho-Kho.
132
102
36
```

1. What is the output of the following code?

```
Class Employee {
```

```
 Void disp(char c){

 System.out.print("Employee name starts with :" +c+ ".");

 System.out.print("His experience is : 11 years. ");

 }
}
```

```
Class Main extends Employee {
```

```
 Void disp(Char c) {

 Super.disp(c);
```

```

 System.out.print("Another employee name also starts with :" +c+" .");
 new Employee().disp("D");
 disp(7);
 }

String disp (int c) {
 System.out.print("His experience is :" +c+" ");
 return "Bye";
}

}

Public class Demo11 {
 Public static void main (String a[]){
 Employee emp = new Main();
 emp.disp("S");
 }
}

```

1. Employee name starts with : S. His experience is : 11 years. Another employee name also starts with : S. Employee name starts with : D. His experience is : 11 years. His experience is : 7.
2. Employee name starts with : S. His experience is : 11 years. Another employee name also starts with : S. His experience is 7 years
3. Employee name starts with : S. His experience is : 11 years. Another employee name also starts with : S. Employee name starts with : D. His experience is
4. Employee name starts with : S. His experience is : 11 years. Another employee name also starts with : S.

Predict the output of below code

```

public class Test {

 public static void main (String a[]){
 System.out.print("Implementing type Casting");
 Dog d = new Dog();
 BullDog bd = (BullDog) d;
 bd.show();
 }
}

```

```

class Dog{
 void show(){
System.out.print("Dog");
}
}
class Cat{
 void show(){
System.out.print("Cat");
}
}
class BullDog extends Dog{
 void show(){
System.out.print("BullDog");
}
}

```

Ans tricky as runtime error and Implementing type casting also comes. But I guess more appropriate is runtime error

RUNTIME ERROR : java.lang.ClassCastException

```

Implementing type CastingException in thread "main" java.lang.ClassCastException:
certificationJava.Dog cannot be cast to certificationJava.BullDog at
certificationJava.Dem011.main(Dem011.java:9)

```

Which code fragment can be inserted at Line 26 to display the output as “Inside catch(RuntimeException) finally end” ?

```

public class Dem011 {

 public static void main (String a[]){

 {
 try
 {
 method();
 System.out.print("Inside try");
 }
 catch (RuntimeException ex)
 {

```

```
 System.out.print("Inside catch(RuntimeException)");
 }
 catch (Exception ex1)
 {
 System.out.print("Inside catch(Exception)");
 }
 finally
 {
 System.out.print("finally");
 }
 System.out.print("end");
}
}

public static void method()
{
 //Line 26
}
}
```

Ans: Option A

**throw new RuntimeException();**

Predict the output of the foll

Square.java

```
final class Square {
 private double length, breadth;
 Square(double length, double breadth) {
 this.length= length;
 this.breadth= breadth;
```

```

}

Square(Square square){
 System.out.println("Copy Constructor Invoked");
 length =square.length;
 breadth= square.breadth;
}
public String toString() {
 return "(" + length +"+"+breadth+ ")";
}

Main.java

class Main{

 public static void main(String args[]){
 Square square1= new Square(110,115);
 Square square2= new Square(square1);
 System.out.println(square2);
 }
}

```

Ans: Option A

Copy Constructor Invoked  
(110.0+115.0)

Which Line fragment can be inserted in Line 1 will help to get the output as 110231(choose all apply)

```

package certificationJava;

public class InnerClassDemo {

 private int bookid=110;

 class Book
 {

```

```

private int bookid=231;
private int getBookid()
{
 return bookid;
}

public void main (String [] args)
{
 Book book = new Book();
 System.out.println(book.getBookid());
}
}

private int getBookid() {

 return bookid;
//Line1

}

}

```

**Ans : Option 2 and Option 4**

```

InnerClassDemo innerClassDemo = new InnerClassDemo();
InnerClassDemo.Book book = innerClassDemo.new Book();
System.out.printf("%d",innerClassDemo.getBookid());
book.main(args);

```

and

```

InnerClassDemo innerClassDemo = new InnerClassDemo();
Book book = innerClassDemo.new Book();
System.out.printf("%d",innerClassDemo.getBookid());
book.main(args);

```

Which of the below option **fails** at Line 7(choose all that apply)

Employee.java

```
public class Employee {
```

```

static final int empid =1101;

}

SuperDemo.class

class Unit extends Employee{

 int empid =1102;

 void display()
 {
 //Line7
 }
}
class SuperDemo {
 public static void main(String [] args)
 {

 Unit unit = new Unit();
 unit.display();

 }
}

```

### Options

- i. System.out.println("Maximum Speed"+super.empid);
- ii. System.out.println("Maximum Speed"+ new Employee().empid);
- iii. Employee emp1 = new Employee();
 System.out.println("Maximum Speed"+ new Unit().empid);
- iv. System.out.println("Maximum Speed"+ Employee.empid);

All 4 options works fine. Not sure if question is wrong or what

Given

The below code fragment can be inserted at Line 1 and Line 2.What will be the output?

```
ConstructorDemo1 constructorDemo1=new ConstructorDemo1(1101,"Jacklin");
ConstructorDemo1 constructorDemo2=new ConstructorDemo1(1102,"John",25);

class ConstructorDemo1 {
 private int id;
 private final String name;
 static final int age=22;
 ConstructorDemo1(int i,String n){
 id=i;
 name=n;
 }
 ConstructorDemo1(int i,String n,int a){
 id=i;
 name=n;
 }
 void display(){
 System.out.println(id+" "+name+" "+age);
 }
 public static void main(String args[]){
 //Line1
 //Line2
 constructorDemo1.display();
 constructorDemo2.display();
 }
}
```

**Ans Option B**

1101 Jacklin 22  
1102 John 22

Predict the output of the below code

```
public class InnerClassDemo {

 InnerClassDemo()
 {
 System.out.print("InnerClassDemo Constructor");
 }
}

Demo.java

class Demo {

 Demo()
 {
 System.out.println("Demo Constructor");
 }
 public void disp()
 {
 System.out.print("Simple Class");
 }

 public static void main(String[] args)
 {
 InnerClassDemo innerClassDemo=new InnerClassDemo();
 innerClassDemo.createDemo();
 }
 void createDemo()
 {
 (new Demo() {}).disp();
 }
}
```

**Ans : Option A : Compilation fails**

Predict the output

```
class Main{

 public void display(int i) {
 System.out.println("Inside First");
 }
 public void method(int i,int j) {
 System.out.println("Inside Second");
 }
 public void method(int... k) {
 System.out.println("Inside Third");
 }

 public static void main(String args[]){
 new Main().method(110);
 new Main().method(110,210);
 new Main().method(110,210,310); //Line1
 new Main().method(110110,210,310,410); //Line2
 }
}
```

Ans:

```
Inside Third
Inside Second
Inside Third
Inside Third
```

**Predict the output**

```
public class Book {
 int bookid = 2356;
}

public class Book1 extends Book{
 int bookid = 1167;

}

public class Book2 extends Book1
{
 int bookid = 2378;
 void display()
 {
 System.out.println(super.super.bookid);//Line10
 System.out.println(super.bookid);//Line11
 System.out.println(bookid);
 }
}

class Demo {

 public static void main(String[] args)
 {
 Book2 book2 = new Book2();
 book2.display();
 }
}
```

**Ans: A. Compilation Fails because of an error in Line10**

When the following code is inserted in Line 6. What's the output

```
Apple apple = (Apple)typeCastDemo;
class Apple {
}
public class TypeCastDemo {
 public static void main(String[] args)
 {
 TypeCastDemo typeCastDemo = new TypeCastDemo();
 //Line6
 }
}
```

**Ans Option C.**

**Compilation fails as typecast can't be done from TypecastDemo to Apple**

What code fragment can be inserted at Line3 to enable the code to print 188.22

```
enum Fruits{
 APPLE,
 MANGO,
 STRAWBERRY,
 LICHI;

 double calculate(double a, double b) {
 switch(this) {
 case APPLE:
 return a+b;
 case MANGO:
 return a-b;
 case STRAWBERRY:
 return a*b;
 case LICHI:
 return a/b;
 default :
 throw new AssertionError("Unknown input"+this);
 }
 }

}

public class EnumDemo {
 public static void main(String[] args)
 {
 //Line3
 }
}
```

**Ans : Option A**

```
double res = Fruits.MANGO.claculate(298, 109.78);
```

Whats the output

```
public class Parent {
 void message()
 {
 System.out.println("Inside parent class");
 }
}
public class Derived extends Parent{

 void message()
 {
 System.out.println("Inside derived class");
 }
 void display()
 {
 message();
 super.message(); //Line1
 }
}

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

 {
 Derived derived=new Derived();
 derived.display(); //Line2
 }
 }
}
```

```
 }
 }
```

**Ans: Option D.**  
**Inside derived class**  
**Inside parent class**

What is the result of attempting to compile and run this program

```
public class Bank extends Exception{

}
public class Customer extends Bank {

}
public class ExceptionDemo {

 public static void main(String args[]) {

 try
 {
 throw new Customer();
 }

 catch (Bank bank) {

 System.out.println("Bank catch Block");
 }

 catch(Customer customer) {

 System.out.println("Customer catch Block");
 }
 }
}
```

```
 }
 }
}
```

**Ans : Option C. Compilation error because Customer class exception is not throwable.**

**Whats the output of the following**

```
public class Demo extends Book {
 int bookid = 4567;

 public int getValue() {
 return bookid;
 }

 public void call() {
 System.out.println(getValue());
 System.out.println(super.getValue());

 }

 public static void main(String args[]){
 Book book = new Book();
 book.call(); //Line3
 super.call(); //Line4
 }
}
```

Book.java

```
public class Book {
 int bookid = 17897;

 public int getValue() {
 return bookid;
 }

}
```

**Ans: Option D. Compilation fails because of an error in line 3 and line 4**

2. What is the output of the following code ?

Package exceptions;

```
Import java.io.*;
Public class ExceptionDemo{
 Static class Car implements AutoCloseable{
```

```

 Public void close(){
 System.out.print("Automatic Door Close");
 }
}

Static class carWindow implements Closerable{
 Public void close(){
 System.out.print("CarWindow");
 throw new RuntimeException();
 }
}
}

Public static void main(String[] args){
 Try(Car car=new Car();
 CarWindow carWindow=new CarWindow()){
 System.out.print("Inside try block");
 }
 Catch(Exception e){
 System.out.print("Inside catch block");
 }
 Finally{
 System.out.print("finally");
 }
}
}
}

```

- a. Automatic Door close CarWindow Inside try block inside catch blockfinally
- b. Automatic Door Close CarWindow Inside catch blockfinally
- c. Inside try blockCarWindowAutomatic Door CloseInside catch blockfinally
- d. An exception is thrown at run time
- e. Compilation fails

**Ans : c. Inside try blockCarWindowAutomatic Door CloseInside catch blockfinally**

60. Given:

```

Public class ExceptionDemo1{
Static class Car implements AutoCloseable{
 Public void close(){
 System.out.print("Car door close");
 Throw new RuntimeException();
 }
}
Static class CarWindow implements Closeable{
 Public void close(){
 System.out.println("Car window close");
 }
}

```

```

Throw new RuntimeException()
}
}
Public static void main(String[] args){
Try{
 //Line 1
}
Catch(Exception e){
System.out.println("Catch exception");
}
Finally{
System.out.print("finally");
}
}

```

Which of the below code can be inserted at Line1 to display THE OUTPUT AS “try block finally” (Choose all that apply)

A)Car car=new Car();  
**CarWindow carWindow=new CarWindow();**  
**System.out.print("try block");**

B)Car car=new Car();  
**System.out.print("try block");**

C)Car car=new CarWindow()  
**System.out.print("try block");**

D)**System.out.print("try block")**

. Which two statements are true for a two-dimensional array?

- A.It is implemented as an array of the specified element type
- B.Using a row by column convention, each row of a two-dimensional array must be of same size
- C.At declaration time, the number of elements of the array in each dimension must be specified
- D.All the methods of the class Object may be invoked on the two-dimensional array

- a) Option (A) and (B)
- b) Option (A) and (B)
- c) Option (B) and(C)
- d) Option (C) and (D)**

Which of the below code fragment needs to be inserted at Line12 to display the output as 15.

```
public class ExceptionInClass {
 int data=10 ;
 void calculate() throws Exception
 {
 try
 {
 data++;
 try
 {
 data++;
 // Line12
 }

 catch(Exception ex)
 {
 data++;
 }
 catch(Exception ex)
 {
 data++;
 }
 }
 }

 void display()
 {
 System.out.println(data);
 }
 public static void main(String[] args) throws Exception
 {
 ExceptionInClass exceptionInClass = new
ExceptionInClass();
 exceptionInClass.calculate();
 exceptionInClass.display();
 }
}
```

**Ans: None of the above. If not write answer as option A**

What is the output when the below code is compiled and executed?

```
import java.util.regex.Matcher;
import java.util.regex.Pattern;

public class Demo1{
 public static void main(String[] args) {
```

```

 ("x*y");
 Matcher match=pattern.matcher("y");
 Boolean boolean1=match.matches();
 System.out.println(boolean1);
 }
}

```

**Ans: Option A True**

Output of below code is

```

class Light{
 Boolean isOn;

 void turnOn(){
 isOn=true;
 }

 void turnoff(){
 isOn=false;
 }

}

class LightDemo{
 public static void main(String[]args){
 Light light1=new Light();
 Light light2=new Light();
 light1.turnOn();
 System.out.println("light1 is on");
 light1.turnoff();
 System.out.println(light1.isOn);
 System.out.println(light2.isOn);
 }
}

```

**Ans : True**

**False**

**Null**

What will be the output of the code given below?

```

public class ABC{

 public static void main(String[]args){
 Boolean flag=false;
 if (flag = true){
 System.out.println("true");
 if (flag = false)
 {
 System.out.println("false");
 }
 }
}

```

```
}}}
```

**Ans Option A. True**

What will be the output of the following code

```
public class Test{
 public void method()
 {
 for(int i=0;i<3;i++) {
 System.out.println(i);
 }
 }
 public static void main(String[] args){
 method();
 }
}
```

**Ans: C. Compilation Fails**

What error do we get when we compile the below code

```
public class Main{

 static int[] x;
 static{
 x[0]=102;
 }
 public static void main(String[] args){
 System.out.println(x);
 }
}
```

**Ans java.lang.ExceptionInInitializerError**

What is the output for the below code

```
public class VarArgsDemo{

 static void func(int...x)
 {
 System.out.println("Number of arguments "+x.length);

 for(int i:x)
 }
```

```

 System.out.print(i+"");
 System.out.println();
 }
 void func(int a) //Line1
 { System.out.println("one"); }

 public static void main(String[]args){
 new VarArgsDemo().func(150);
 func(11, 12, 13, 14);
 func(); }
 }

```

**Ans:Option B**

**one**

**Number of arguments 4**

**11121314**

**Number of arguments 0**

How many objects are eligible for garbage collection after executing line8.

```

public class Employee {

 double salary;
 public static void main(String[]args){
 Employee employee1=null;
 Employee employee2=null;
 employee1= new Employee();
 employee2= new Employee();
 employee1= new Employee();
 Employee employee3=null;
 employee1= employee3=null; //Line8
 System.out.println("Hello World");
 }
}

```

**Ans :3 objects**

Which statements are true about the following code snippet?(choose all that apply)

```

Public class Developer{}
Public class Employee{
 Public String empName;
}

```

```

Public class Tester extends Employee{
 Public Developer developer;
}

```

```

Public class Testing extends Tester{}

```

- a. Testing has a empName
- b. Testing has a Developer
- c. Testing is a Developer
- d. Testing is a Employee**
- e. Tester is a Testing
- f. Employee has a Developer

Ans : d & a

Observe the below code snippet:

```
public class Tree extends BasePlant{
 public void growFruit(){}
 public void dropLeaves(){}
}

public class BasePlant {
 private String name;
 public BasePlant(String name){
 this.name=name;
 }
 public String getName(){
 return name;
 }
}
```

Which of the following statement is true?(choose all that apply)

- a. The code will be compiled without any changes
- b. The code will be compiled only if the below code is added to the Tree class**  
Public Tree() {super("Plant");}
- c. The code will be compiled only if the below code is added to the BasePlant class  
Public BasePlant() {Tree();}
- d. The code will be compiled only if the below code is added to the BasePlant class**  
Public BasePlant() {this("Plant");}

Ans : b & d

Predict the output

Apple.java

```
public class Apple {
 public void color(){
 System.out.println("Red");
 }
}
```

Mango. Java

```
public class Mango extends Apple{
 @Override
 public void color(){
 System.out.println("Yellow");
 }
 public static void main(String[] args){
 Apple apple=new Mango(); //Line1
 apple.color(); //Line2
 }
}
```

**Ans: Yellow**

Given:

```
public interface interfaceDemo{
 //Line1
}
```

Select the suitable code fragment can be inserted at Line1(choose all that apply)

- a. **void display(int x);**
- b. **void display(int x) {}**
- c. **public static void display(int x){}**
- d. **default void display(int x);**
- e. **public interface Demo{}**

**Ans : a,c and e**

Analyze the code and select the suitable outcome

```
class Apple {

 private Apple() {

 System.out.println("Apple constructor");
 }

 void display(){
 System.out.println("Apple constructor");
 }
}

public class Main {

 public static void main(String[] args){
 Apple apple=new Apple(); //Line2
 apple.display();
 }
}
```

**Ans: Option D: Unresolved compilation problem: The constructor Apple() is not visible**

Output of the below code

```
public class Demo {
 static int x=232;
 int y=135;
 public void display(){
 System.out.print("Inside Demo");}
 public static void staticMethod(){
 System.out.print(x); //Line 8
 }

 public static void main(String[]args)
 {
 Demo.staticMethod(); //Line 13
 Demo demo=new Demo();
 demo.staticMethod(); //Line15
 staticMethod();
 demo.display(); //Line 16
 }
}
```

```
}
```

**Ans : Option B**

**232232232Inside Demo**

**Check line13. If its not Demo.staticMethod(); then answer will be 232232Inside Demo**

Output of the below code

```
public class Demo {

 public static void main(String[] args){
 try{
 throw 110;
 }

 catch(int ex){
 System.out.println("Caught Exception" + ex);
 }
 }
}
```

**Ans: Compilation Fails**

Predict the output of the below

```
public class TestDemo {

 public static void main(String[] args){
 int sum, a=10, b=10;
 try{
 System.out.println(sum=a/b);
 return; //Line 1
 } catch(ArithmetcException | Exception e){ //Line2
 System.out.println(e.getMessage());
 }finally{
 System.out.println("in finally");
 }
 }
}
```

**Ans: Compilation fails because of the error in Line2**

What are the different types of memory areas used by JVM(choose two)?

1.Class

**2.Heap**

**3.Stack**

4.Queue

JVM in java is a

1.Debugger

2Assembler

**3.compiler**

4.Interpreter

132. What is magic number in java in the context of java programming best practices?

1.A number which gets printed on the console

**2.A direct usage of the number int the code**

3.A number which magically disappears from the code

4.A number which is generated through error

What is the output of the below code

```
public class person {
```

```
 public person(String name){
 System.out.println(name);
 }
```

```

}

public class student extends person{

 public student(){ //Line 8
 System.out.println(" Student");
 }

 public static void main(String[] args) { // Line 11
 new person("Jacklin");
 }
}

```

**Ans: Compilation fails because of an error in Line 8**

3. Given:

```

public abstract class Employee {

private int empld;

private int salary;

public abstract void display();

public void setValues(int empld, int salary){

this.empld = empld;

this.salary = salary;

}

}

```

Which of the following classes provide the right representation of the child class of Employee class?

- 1) public abstract class Child extends Employee {
 private int z;
 }
- 2) public class Child implements Employee {
 private int z;
 }
- 3) public class Child extends Employee {
 private int z;
 public void display();
 }

4) public class Child extends Employee {  
    private int z;  
    public void display() {  
        /\* code here \*/  
    }  
}

Ans : 4) public class Child extends Employee {

```
 private int z;
 public void display() {
 /* code here */
 }
```

4. Given an abstract Class Customer as below:

```
public abstract class Customer
{
 public abstract String getCustomerType();
}
```

Select a Valid implementation of getCustomer Type method in another class, from the below options:

1) abstract class C1 extends Customer{  
    public string getCustomer Type()  
    { return “Premium”;  
    }  
}

2) Customer customer = new Customer(){  
    public String getCustomerType()  
    { return “Premium”;  
    }  
}

3) class C1 extends Customer{  
    public String getCustomerType()  
    { return “Premium”;  
    }  
}

4) new Customer(){  
    public String getCustomerType()  
    { return “Premium”;

```
 }
}
```

Ans : 3) class C1 extends Customer{

```
public String getCustomerType()
{ return "Premium";
}
}
```

Output of the following

```
class Customer{
int customerId = 11201;
Customer() {
 customerId = 11240;
}
}
class Main {
public static void main(String args[]){
 Customer customer = new Customer();
 System.out.println(customer.customerId);
}
}
```

Ans: Option B 11240

Code to be written to get the output as below

False

Simple

Demo

For

Regular

Expressions

Using

Pattern

Matching

```
public class RegExDemo {
 public static final String string1="Simple demo for +"regular expressions"+
 usingpatternmatching";
 public static void main(String[] args){
 //Line 1
 //Line2
 }
}
```

**Ans: Option 1**

```
System.out.println(string1.matches("\t"));
String[] splitString=(string1.split(" "+));
//(String1.split(\s+)) not working in my computer so did like this
for(String string: splitString){
 System.out.println(string);
}
System.out.println(string1.replaceAll("\s","\\t"));
}
```

What is the result of attempting to complete and run this program?

```
public class Demo {
 public static void main(String[] args){
 String c="a";//Line 3
 switch(c) {
 //Line4
 case 65 ://Line5
 System.out.println("One");
 break;
 case "a"://Line6
 System.out.println("two");
 case 3://line 10
 System.out.println("three");
 break;
 }
 }
}
```

**Ans:D Computation fails because of an error in Line 5 and Line 10**

Select all possible options that are valid among the following Enums can be defined inside\_\_\_\_\_

- a) An interface
- b) A class {Multiple choice question}
- c) A static Context
- d) A method

Which code fragment can be inserted at Line 1 to enable the code to print as “Number of Days =25”

```
class Demo {
 public static void main(String[] args)
 {
 int monthValue=2;
 int yearValue=4000;
 int numberOfDay=10;

 switch(monthValue) {

 case 1: case 3: case 5:
 case 7: case 8: case 10:
 case 12;
 numberOfDay=31;
 break;
 case 4: case 6:
 case 9: case 11:
 numberOfDay=28;
 break;
 case 2:
//Line1
 numberOfDay=25;
 else
 numberOfDay=28;
 break;

 default:
 System.out.println("Number of Days =" +numberOfDay);
 }
 }
}
```

}

**Ans: Option not clear. Either one can come. Make sure assignment operator is there(==)**

```
if((yearValue% 4 ==0) &&
 (yearValue% 100==0)
 ||(yearValue% 400==0))
```

Or

```
if((yearValue% 4 ==0) ||
 (yearValue% 100==0)
 ||(yearValue% 400==0))
```

81. Identify which of the following class breaks its input into tokens using a whitespace pattern?

- a. InputStreamReader
- b. Console
- c. Scanner
- d. BufferedReader
- e. DataInputStream

What is the output when below code is compiled and executed.

```
import java.util.regex.Pattern;
public class RegExDemo2 {
 private static final String String1="";
 private static final String String2="one two three four five";
 public static void main (String[] args){
 Pattern pattern=Pattern.compile(String1); //Line 7
 String[] strArr=pattern.split(String2); //Line 8
 for(String str:strArr){
 System.out.println(str);
 }
 }
}
```

**Ans : Option C (if pattern.compile and pattern.spilt don't have dot means then compilation error)**

- C)one
- two
- three
- four
- five

What is the output of the below code

```
package certificationJava;

public class ABC {
 public static void main(String args[]){
 Boolean flag=false;
 if(flag=true){
 System.out.println("true");
 }
 if(flag==false){
 System.out.println("false");
 }
 }
}
```

**Ans: Option A . True**

Which is the correct code fragment to be inserted at Line 1 to execute the code to print count starts from 111,112,113....

```
public class Demo2 {

 public static void main(String[] args){
 int[]X={111,112,113,114,115,116,117,118,119,110};
 //Line1
 System.out.println("count is"+i);
 }
}
```

**Ans : Option B**

**for(int i:X){**

Predict the output of the below

```
public class Calculator {

 int a=123;
 int b=200;
 public void display(){
 System.out.println("a"+a+"b"+b+"");
 }
}

public class CalculatorDemo {
 public static void main(String[] args)
 {
 Calculator calculator1=new Calculator(); //Line1
 Calculator calculator2= Calculator1//Line2
 calculator1.a+=1;
 calculator1.b+=1;
 System.out.println("calculator1 values");
 calculator1.display();
 System.out.println("calculator2 values");
 calculator2.display();
 }
}
```

**Ans : D. Compilation fails because of error in Line2**

Output of the following

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

 int i1=0;
 int[] j={11,111,14,19,116,215}; //line4
 for (int i1:j) //line5
 System.out.printf("%d",i1);
 }
}
```

**Ans :Option C: compilation fail because of an error in line5**

Output of the following

```
abstract class Customer {

 public int custId;
 Customer()
 {
 custId=23456;
 }
 abstract public void setId();
 abstract final public void getid(); //Line11
}

class Demo extends Customer{
 public void setId(int custId)
 {
 this.custId=custId;
 }
 final public void getid() //Line9
 {
 System.out.println("Customerid"+custId);
 }
 public static void main(String[] args)
 {
 Demo demo=new Demo();
 demo.setId(1102);
 demo.getid();
 }
}
```

Ans :

- a) compilation fails because of an error in Line9
- b) compilation fails because of an error in Line11

Output of the below code

```
public class Employee {

 public final void show(){
 System.out.println("show()inside Employee");
 }
}

final class Unit extends Employee{
 public void show1(){ //Line1
 final int x=100;
 System.out.println("show()inside Unit");
 System.out.println(x);
 }
}

public class Demo11{
 public static void main(String[] args){
 Employee employee=new Unit();
 new Unit().show1();
 }
}
```

**Ans:Option D**

**show()inside Unit  
100**

Given

```
Class Parent{
}
Class Child extends Parent{
}
Final class GrandChild extends Child{
}
```

Which of the following statement is not true about the above code?

- a) The above code represents the multi-level inheritance with the two level
- b) The GrandChild class can Access the protected and public members of the parent and child class
- c) **Instance of parent class can accept the reference of the child class but not the reference of GrandChild class**
- d) The GrandChild class can override the methods of both Parent class and Child class

In the below code snippet identify which of the following method compares the given values and return an int which tells lesser or greater

```
public class WrapperClassDemo {

 public static void main(String aa[]){
 int x=90;
 Integer i1=new Integer(x);
 int y=90;
 Integer i2=new Integer(y);

 System.out.print(i1.compareTo(i2)+" "+Integer.compare(i2,i1)+" "+i1.equals(i2)+"
 "+(i1==i2));
 }
}
```

- a) Compare()
- b) Equals()
- c) compareTo()
- d) ==

Output of the below code

```
public class TestDemo {
 public static void main(String[] args){
 Integer n1=new Integer(100);
 Integer n2=new Integer(100);
 Integer n3=127;
 Integer n4=127;
 Integer n5=128;
 Integer n6=128;
 int n7=129;
 int n8=129;
 System.out.print(n1==n2);
 System.out.print(n3==n4);
 System.out.print(n5==n6);
 System.out.print(n7==n8);
 }
}
```

**Ans A: false true false false**

Output of the following

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

 TreeSet<String> treeset=new TreeSet<String>();
 treeset.add("first");
 treeset.add("First");
 treeset.add("Second");
 System.out.println(treeset.ceiling("Fir"));
}
```

**Ans : Option C**

**First**

**If <String> not defined then compilation error comes**

Output of the following

```
public class TestDemo {
```

```

public static void main(String[] args){
ArrayList Strings=new ArrayList();
Strings.add("aAaA");
Strings.add("AaA");
Strings.add("aAa");
Strings.add("AAaa");
Collections.sort(Strings);
for(string:Strings){
System.out.print(string);
}}}

```

**Ans: Option A Compilation Fails**

Output of the following

```

public class person {
 private final String name;

 public person(String name){
 this.name=name;
 }

 public String toString(){
 return name;
 }
}

import java.util.TreeSet;
public class Group extends TreeSet {

 public static void main(String[] args){
 Group g=new Group();
 g.add(new person("Hans"));
 g.add(new person("Jane"));
 g.add(new person("Hans"));
 System.out.println("Total"+g.size());
 }

 public boolean add(Object o){
 System.out.println("Adding"+o);
 return super.add(o);
 }
}

```

**Ans : Option A**

a) Adding Hans

An exception is thrown at the runtime(java.lang.ClassCastException)

Output of the following

```

public interface StaticInterface {

```

```

 static void staticMethod()
 {
 System.out.println("Inside interface");
 }
}

public class StaticInterfaceImpl implements StaticInterface {

 public void staticMethod()
 {
 System.out.println("Inside class");
 }
}

public class StaticDemo {

 public static void main(String[] args)
 {
 new StaticInterfaceImpl().staticMethod();
 }
}

```

**Ans: Option D. code will print “inside class” on execution**

**Inside class**

Output of the following

```

import java.time.LocalDate;
import java.time.format.DateTimeFormatter;

public class Formatting {
 public static void main(String[] args)
 {
 LocalDate date=LocalDate.of(2016,11,13);

 DateTimeFormatter formatter= DateTimeFormatter.ofPattern("dd/MMM/YYYY");
 System.out.println(date.format(formatter));
 }
}

```

**Ans: Option D: 13/Nov/2016 will be printed**

**13/Nov/2016**

Predict the output

```

public interface Interface1 {

 default void method1()
}

```

```

 {
 System.out.println("Inside default method");
 }

 }

public interface DefaultExtends extends Interface1{
 default void method1()
 {
 System.out.println("Default method redefined");
 }
}

public class InterfaceWithDefaultMethod implements DefaultExtends{
 public static void main(String[] args)
 {
 InterfaceWithDefaultMethod defaultExtend=new
 InterfaceWithDefaultMethod(); //Line4
 defaultExtend.method1(); //Line5
 }
}

```

**Ans: Option B**

**Default method redefined**

What happens when default keyword is removed from the below code snippet

```

public interface Interface1 {

 default void method1()
 {
 System.out.println("Inside default method");
 }
}

```

a.method cannot be overridden in the implementing classes

b.method can be overridden in the implementing classes

c.method cannot be given body in the interface

d.compilation error occurs

a) a and b

b) a,b and c

**c) c and d**

d) b and c

Select the valid code fragment according to java coding standard?

- 1) public void draw(String s){
 if(s.equals("Square")){

```

drawSquare();
}
if(s.equals("Rectangle")){
drawRectangle();
}
}

2) public void draw(String s){
if("Square".equals(s){
drawSquare()
}
if("Rectangle".equals(s)){
drawRectangle();
}
}
}

only option(1) is valid
only option(2) is valid
Both(1) and (2) are valid
Both(1) and (2) are invalid

```

Whats the output of the below

```

public class Ex1 {
public String formatInput(String i){
if(i.trim().length()==9){
StringBuilder s1=new StringBuilder();
s1=s1.insert(0, "+1(");
s1=s1.insert(6, ")");
s1=s1.insert(10, "-");
return s1.toString();
}
return null;
}
public static void main(String args[]){
Ex1 ob=new Ex1();
String I;
ob.formatInput(I);
}
}

```

- a) compilation fails at Line3
- b) Compilation fails at Line 6
- c) Null pointer exception will be thrown if the value of I is null

- d) Compilation fails due to error in Line7

```
// I got below compilation error
```

```
Exception in thread "main" java.lang.Error: Unresolved compilation problem:
```

Consider the below code snippet

```
Locate locate=new Locate("da","DK");
NumberFormat numberFormat=NumberFormat.getInstance(locate);
String number=numberFormat.format(100,99);
System.out.println(number);
```

Here NumberFormat.getInstance() follows which design pattern?

- a) Factory method pattern
- b) Singleton pattern
- c) Abstract Factory pattern
- d) Builder pattern

Output of the following

```
public class TestDemo {

 static int a=0;
 static ArrayList b;

 @BeforeClass
 public static void beforeClass(){
 a=10;
 b=new ArrayList();
 }
 @BeforeMethod
 public void int1(){
 a=15;
 b.add(a);
 }
 @Test
 public void test(){}
```

```

 a=a+20;
 System.out.print(a);
 System.out.println(b);
}
@Test
public void test1(){
 a=a+30;
 System.out.print(a);
 System.out.print(b);
}
}

```

Predict the output?

- a) 35[15]  
45[15,15]
- b) 35[15]  
65[15,15]
- c) 35[15]  
45[15]
- d) 35[15]  
65[15]
- e) 35[15]  
65[30]

Output of the below code snippet

```

public class Test {

 public static void main(String[] args){
 int [][] x;
 x=new int[3][4];
 for(int i=0;i<3;i+=2){
 for(int j=0;j<4;j++){
 x[i][j]=i+j;
 System.out.print(x[i][j] + " ");
 }
 }
 }
}

```

**Ans : Option B**

**01232345**

Output of the foll

```
public class Pet {
 public void displayName(){
 System.out.println("Inside Pet");
 }
}

public class Dog extends Pet{

 public void displayName(){
 System.out.println("Inside Dog");
 }
}

class Demo{
 public static void main(String[] args){
 Pet pet=new Dog();
 pet.displayName();
 }
}
```

**Ans Option D: Compilation Fails.**

Output of the following

```
public class Hello {
 public static void main(String[] args){
 String s="How\"are\"you?";
 System.out.println(s);
 }
}
```

**Ans: Option A**

**How"are"you?**

Output of the following

```
public class WrapperClassDemo {
```

```

public static void main(String args[]) {

 Integer intWrapper=Integer.valueOf("12345");
 Integer intWrapper2=Integer.valueOf("11",2);
 Integer intWrapper3=Integer.valueOf("E",16);
 System.out.println(intWrapper+" "+intWrapper2+" "+intWrapper3);
}

}

```

**Ans: Option C**

**12345 3 14**

Predict the output of the following

```

public class Demo11{
 public static void main(String args[]) {
 Set numbers=new HashSet();
 numbers.add(new Integer(45));
 numbers.add(88);
 numbers.add(new Integer(77));
 numbers.add(null);
 numbers.add(789L);
 Iterator iterator=numbers.iterator();
 while(iterator.hasNext())
 System.out.print(iterator.next());
 }
}

```

**Ans: Option F**

**null1789884577**

Which of the below code has to be inserted at Line1, to sort the keys in the props HashMap variable?

```

public class Demo11{
 public static void main(String args[]) {
 HashMap props=new HashMap();
 props.put("key45","some value");
 props.put("key12","some other value");
 props.put("key39","yet another value");
 Set s=props.keySet();
 //Line1
 }
}

```

**Ans: Option B. Collections.sort(s);**

Output of the following

```

public static Collection get(){
 Collection sorted=new LinkedList();
 sorted.add("B");
 sorted.add("C");
 sorted.add("A");
 return sorted;
}
public static void main(String[] args){
 for(Object obj: get()){
 System.out.print(obj+".");
 }
}

```

**Ans : Option B**  
**B.C.A.**

Output of the following

```

public static void main(String[] args){
 TreeSet tset=new TreeSet();
 tset.add(new item());
 TreeSet b=tset;
}

```

**Ans : Option A. Compilation Fails.**

Which of the following code snippet can be inserted at line1 to display the output as

76  
Hello

```
class Apple<A> {
```

```

A obj;
Apple(A obj)
{this.obj=obj;
}

public A getObject()
{return this.obj;

} }
class Main{
 public static void main(String[] args){
 //Line1
}
}

```

**Ans :Option A and Option B**

```

Apple apple=new Apple(76);
System.out.println(apple.getObject());
Apple appleObj=new Apple("Hello");
System.out.println(appleObj.getObject());

```

82. Refer the below code snippets and predict the outcome?

```

Public class RepeatingAnnotations{
 @Retention(RetentionPolicy.RUNTIME)
 public @interface Chocolates{
 Favourite[] value() default();
 }

```

```

 @Favourite("Diary Milk")
 @Favourite("Kit Kat")
 @Favourite("5 star")
 @Favourite("Galaxy")
 public interface Chocolate{
 }

```

```

 @Repeatable(value=Chocolates class)
 Public @interface Favourite{

```

```

 String value();
 }

```

```

 Public static void main(String[] args){

```

```

Favourite[] a=Chocolate.class.getAnnotationsByType(Favourite.class);
Chocolates chocolates=Chocolate.class.getAnnotation(Chocolates.class); //Line5
for(Favourite favourite: chocolates.value()){
 System.out.println(favourite.value()); } }

```

- a. Nothing will be displayed
- b. null will be printed
- c. Runtime exception will be thrown at Line 5
- d. **Dairy Milk**
- Kit Kat**
- 5 Star**
- Galaxy**

Output for the following

```

public class RepeatingAnnotations {
 @SuppressWarnings("all") //line1
 @SuppressWarnings("deprecation") //line2
 public void over()
 {
 new Date().setDate(00); } }

```

**Ans: Option B.**

**Compilation will not be successful as @SuppressWarnings annotation is non-repeatable in nature**

Output of the following

```

public class TestDemo {
 public static void main(String[] args){
 LocalDateTime date1=LocalDateTime.of(2017, Month.FEBRUARY,
11, 15, 30); //Line1
 LocalDateTime date2=LocalDateTime.of(2017, 2, 12, 10, 20);

 System.out.println(date1.compareTo(date2));
 }
}

```

**Ans Option A -1 will be printed as execution result**

Predict the output

```
class Apple {
 int quantity;
}

class Main{
 public static void main(String[] args){
 Apple apple;
 System.out.println("apple quantity");
 }
}
```

**Ans: Option 5: Apple Quantity**

Output of the following

```
public class TestDemo {
 private static Object staticObject;
 public static Object createStaticObject(){
 if(staticObject==null){
 staticObject=new Object();
 }
 return staticObject;
 }
}
```

What changes are required in the above code for successful execution?

- 1.The method createStaticObject should be synchronized
- 2.The method createStaticObject should be private
- 3.The staticObject reference should not be static
- 4.The method createStaticObject should not return Object type

Output of the following

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

```

{
LocalDate date=LocalDate.of(12,11,2017);
System.out.print(date);
}
}

```

Ans: Option D: Exception will be raised as date not in range

Output of the following

```

public class Demo
{
 public void division(int x,int y){
 try{
 int z=x/y;
 }
 catch(Exception e){
 System.out.print("Arithmetic Exception");
 }
 finally{
 System.out.print("finally block");
 }
 }
 public static void main(String[] args)
 {
 Demo demo=new Demo();
 demo.division(0,8);
 }
}

```

**Ans: Option 2. finally block**

Output of the below code

```

public class Demo
{
 void display()
 {
 System.out.println("x= "+x+" y= "+y);
 }

 public static void main(String[] args)
 {
 Demo thisDemo=new Demo();
 thisDemo.get().display();
 }
}

```

**Ans: C . Compilation Fails**

Output of the following

```
public class TestDemo {
 static void myCode() throws MyException{
 try{
 throw new MyException("Test exception");
 }
 catch(Error|Exception ex){
 System.out.print("Inside Error and Exception");
 }
 }

 public static void main(String[] args) throws MyException{
 try{
 myCode();
 }
 catch(Exception ex){
 System.out.print("Inside Exception");
 }
 }
}
```

**Ans Option A**

**Prints Inside Error and Exception**

Output of the following

```
public class ThisDemo
{

 int x;
 int y;
 ThisDemo(){
 x=45;
 y=56;
 }
 ThisDemo get() //Line1
 {
 return this;
 }

 void display()
 {
 System.out.printf("x=%d+y=%d", x, y);
 }

 public static void main(String[] args)
 {
 ThisDemo thisDemo=new ThisDemo();
 thisDemo.get().display();
 }
}
```

Ans: will know answer based on syso only

I got answer as  $x=^*+x+^*y=^*+y$

```
public class student {

 private School school;
 private StudentDetails stuDetails;
 private Fees fees;

 public MarksHistory marksHistory(Marks marksDetails){
 //computation
 }
}
```

**Ans: Lazy Initializtion**

Output of the following

```
public class Demo11{
 public static void main(String[] args){
 Parent obj =new Child();
 obj.display();
 }
}

public class Parent {

 public void display(int a){
 System.out.println("Parent Method");
 }
}

public class Child extends Parent {

 public void display()
 {
 System.out.println("Child Method");
 }
}
```

**Ans: A: Compilation Fails**

Predict the output

```
public class Manager extends Employee {

 public void someManagerMethod(){
 //...
 }
}
```

```

 }
 }

public class Officer extends Employee {
{
 //...
public void someMethod(Employee e){
 Manager m=(Employee)e ; //Line 12
 m.someManagerMethod();
}
}

}

public class Demo {

 public static void main(String s){
 Officer obj=new Officer();
 obj.someMethod(new Officer()); //Line 19
 }
}

```

**Ans: Option 1: Compilation fails because of an error in Line 12**

Output of the following

```

public interface Demo1 {
 public void display(String points);
}

public class Demo2 implements Demo1{
 public void display(String points){};
}

public class Demo3 {
 public Demo1 disp(){
 return null; //more code here
 }
 public String displayValue(){ //Line6
 return null;
 //more code here
 }
}

public class Demo4 extends Demo3{
 public Demo2 disp(){
 //more code here
 return null;
 }
 private String displayValue(){
 //more code here
 }
}

```

**Ans: Option C. compilation of class Demo4 will fail because of an error in line6**

Which of the code segment is written using best practice

1. 

```
List list;
public List getList{
 if(list.size()==0)
 return null;
 else
 return list;
}
```
2. 

```
Integer i1=new Integer(11);
Integer i2=new Integer(11);
System.out.println(i1==i2);
```
3. 

```
String[] str=new String[]{"Hi","Hello","Welcome"};
List strList=Arrays.asList(str);
for(iterator itr=strList.iterator();itr.hasNext()){
 System.out.println(itr.next());
}
```

Ans : Code 2 only is valid

144. //Assume that the first two of three test cases fail in “Testclass”

// Assme all the required import statements are added

```
Public class testrunner{
 Public static void main(String [] args){
 Result result = junitcore.runclasses(testclass.class)
 For (Failure failure : result.getfailures()){

 }
```

```
System.out.println(result.wassuccessful());
}
}
```

- 1) False
- 2) True
- 3) False false true
- 4) False false false

Output of the foll

```
public class collectionsDemo{
 public static void main(String argv[]){
 ArrayList arrList=new ArrayList();
 ArrayList arrListStr=arrList;
 ArrayList arrListBuf=arrList;
 arrListStr.add(1,"SimpleString");//line6
 StringBuffer strBuff=arrListBuf.get(0);//line7
 System.out.println(strBuff.toString());//line8
 }
}
```

**Ans: Option C.Compilation fails because of error in Line7**

Output of the following

```
public class StringTest {

 public static void main(String[] args){
 String joinString=String.join(".", "java", "programming", "course");
 String s1="JAVA", s2="java", s3="Java";
 s1.toLowerCase();
 s3=s3.replace("J", "j");
 System.out.println(joinString);
 System.out.println(s1.equals(s2)+", "+(s2==s3));
 }
}
```

**Ans Option D:**

```
java.programming.course
false, false
```

### **Output of following**

```
public interface DefaultMethodInterface1 {
 default public void defaultMethod(){
 System.out.println("DefaultMethodInterface1");
 }
}

public interface DefaultMethodInterafce2 {
 default public void defaultMethod(){
 System.out.println("DefaultMethodInterface2");
 }
}

public class TestDemo implements DefaultMethodInterface1, DefaultMethodInterafce2{

 public static void main(String[] args){
 DefaultMethodInterface1 defMethln=new TestDemo();
 defMethln.defaultMethod();
 }
}
```

### **Ans: Compilation fails**

- . Which of these statements compile?(choose at least one)

checkbox

1. HashSet hs=new HashSet();

2. HashSet set=new HashSet();

3. List list=new Vector();

```
List values=new HasgSet();
List objects=new ArrayList();
Map hm=new HashMap();
```

### Output of the foll

```
public class TestDemo {
 public static void main(String[] args){
 List list1=new ArrayList();
 list1.add("1");
 list1.add("2");
 list1.add("3");
 List list2=new LinkedList(list1);
 list1.add(list2);
 list2=list1.subList(2,5);
 list2.clear();
 System.out.print(list1+"");
 }
}
```

**Ans: Option 1**

**the program complies successfully and throws exception during runtime**

## Section 1:

1. Which of the following OOP terminology associated with java..... Employee has address

Ans - Inheritance

2. What is the result when the following code is compiled and executed

```
public class Test {
 Long a; //Line1
 long b;
 public Test(long c){
 b=a+c; //Line 2
 System.out.println(b);
 }
 public static void main(String[] args){
 new Test(new Long(10L));
 }
}
```

Ans: Null pointer exception in Line 2 as variable a is not

3. Given

```
class Movie implements Comparator<Integer> {
 public int comparator(Integer o1, Integer o2){
 return o2.compareTo(o1);
 }
 @Override
 public int compare(Integer o1, Integer o2) {
 // TODO Auto-generated method stub
 return 0;
 }
}

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

```

 Integer mov[] = {2019,2017,1989,1994};
 Arrays.sort(mov,new Movie());
 for (int i:mov){
 System.out.print(i+" ");
 }
 }

}

```

Ans – c

2019 2017 1989 1994

4. Identify the output:

```

public class MyDemo {
 public static void main(String[] args){
 int i =5;
 switch(i){
 case 1:
 System.out.println("One");
 break;
 case 2:
 //Line 1
 case 3:
 //Line 2
 System.out.println("Two and Three");
 case 4,5:
 //Line3
 System.out.println("Four and Five");
 break;
 default:
 System.out.println("Default");
 }
 }
}

```

Ans: Compilation error in Line 3 as multiple values are not allowed in case

5. Which of the following is correct usage of a relational operator made in if statement.

Ans : if (firstName.equals("Annie")&&salary==50000)

6. Identify the output of the below code:

```
public class TestDemo {
 public static void main(String[] args){
 boolean a = true;
 boolean b = true;
 boolean c = false;
 boolean d = true;
 System.out.println(a&&b || c&&d);
 }
}
```

Ans - true

7.

```
public class UtilTest {
 @Rule
 public ExpectedException thrown = ExpectedException.none();
 //@Test(expected = Exception.class)
 //Line1
 @Test
 public void test1() throws Exception{
 thrown.expect(NullPointerException.class);
 throw new NullPointerException();
 }
}
```

Ans - @Test

8. Which of the following component is responsible to compile, debug a java program?

Ans-JDK

9. What is the output for the below code?

```
interface Fruits{
 public void printPrice();
}
public class Apple {
 public static void main(String[] args){
 Fruits fruits = new Fruits(){
 public void printPrice(){
 System.out.println("150");
 }
 }
}
```

```
 };
 fruits.printPrice();
}
}
```

Ans-150

10. Which among the following is valid option for wildcards?(select 2 options)

- A. Used to relax restriction on the variable
- B. Used in scenario where type being operated upon is not known
- C. Used in generic method type argument
- D. Can access members of super class

Ans:

- A. Used to relax restriction on the variable
- B. Used in scenario where type being operated upon is not known

11. Which of the below method name is valid as per Java naming convention?

Ans: methodName

12. Consider the Junit test class with junit fixture annotations and the methods as below:

```
@BeforeClass ---- init()
@AfterClass ---- close()
@Before ---- setUp()
@After ---- tearDown()
@Test----testSum1()
@Test----testEven1()
```

In which order the methods will execute?

Ans – init() setup() testSum() tearDown() setUp() testEven() tearDown() close()

13. Which of the following is the correct syntax to declare the abstract method evaluate?

14. Predict the output of the below code.

```

class Car{
 void start(){
 System.out.println("Car Starts");
 }
}
class Bike{
 void start(){
 System.out.println("Bike Starts");
 }
}
class Automobile extends Car{
 void start(){
 System.out.println("Automobile Starts");
 }
}
public class ExceptionDemo {
 public static void main(String[] args){
 System.out.println("Implementing Typecasting");
 Car d = new Car();
 Automobile automobile = (Automobile) d;
 automobile.start();
 }
}

```

Ans :

Displays "Implementing type casting" and RUNTIME EXCEPTION java.lang.ClassCastException

15. Analyze the below code and predict the outcome when compiled and executed?

```

public class Demo extends Book {
 int bookid =4567;

 public int getValue() {
 return bookid;
 }

 public void call() {
 System.out.println(super.getValue()); //Line 1
 }

 public static void main(String args[]){
 Book book = new Book();
 super.call(); //Line 2
 }
}

public class Book {
 int bookId = 17897;
}

```

```

 public int getValue(){
 return bookId;
 }

}

```

Ans - Compilation error in Line2 as super keyword cannot be used in static context

16. Which of the following condition will not allow the finally block to be executed?

Ans – when System.exit(1) is called

17. What is the result of the following?

```

public class TestDemo {
 public static void main(String[] args){
 try{
 throw new ArithmeticException("AE");
 }catch(ArithmaticException e2){
 System.out.println(e2.getClass());
 }catch(Exception e1){
 System.out.println("E1");
 }
 }
}

```

Ans - class java.lang.ArithmaticException

18. What is the result of attempting to compile and run this program?

```

class CustomException extends Exception{}
class Customer extends CustomException{

public class ExceptionDemo1 {
 public static void main(String[] args){
 try{
 throw new Customer();
 }catch (CustomException customException){
 System.out.println("Custom Exception Catch block");
 }catch(Customer customer){
 System.out.println("Customer catch block");
 }
 }
}

```

**Ans – Compilation error because customer class exception is not throwable**

**19. Which of this statement is not correct and will lead to compilation error.....**

**20. What will be the output of the following code when executed?**

```
public class DateTimeTester {
 public static void main(String[] args){
 LocalDateTime localDateTime = LocalDateTime.of(2020,5, 13, 20, 46);

 System.out.println(localDateTime.get(ChronoField.HOUR_OF_DAY)+localDateTime.getDayOfMonth());
 }
}
```

**Ans – 33**

**21. Which of the below code is implemented without best practices standard?**

1. String[] str=new String[]{"Hi","Hello","Welcome"};  
List strList=Arrays.asList(str);  
**for(iterator itr=strList.iterator();itr.hasNext();){**  
System.out.println(itr.next);
  
2. Integer i1=**new** Integer(11);  
Integer i2=**new** Integer(11);  
System.out.println(i1==i2);

**Ans: Option 1 doesnot follow best practices. Can be improved using for(String s:str)**

**22. Which of the following is used for the automatic accurate tracking for the decimal values:**

**Ans:BigDecimal**

23. Given:

```
public class TestDemo1 {
 public static void main(String[] args)
 {
 int i=4;
 int j=4;
 System.out.println(i==j);
 Integer w1=new Integer(4);
 Integer w2=new Integer(4);
 System.out.println(w1==w2);
 }
}
```

Ans: no issues in the above code

JVM is the actual runtime in which a Java application executes.

163: Paul uses interpreter to execute the byte code. The interpreters are platform \_\_\_\_\_

Ans - dependent

164: JVM in Java is a:

Ans --

165: Which of the following is/are true regarding JVM? (Choose 2 options)

Ans --

JVM compiles the bytecode, converts it into native code, and then executes the native code

JVM interprets the bytecode line by line, and then executes it

166:

```
class Account {
 private double balance = 500.00; // member data
 public double getBalance(int x) { // member method
 // logic here
 return balance;
 }

 public static void main(String[] args) {
 Account accnt = new Account(); // object creation
 double value = accnt.getBalance(123456);
 System.out.println("The balance is: " + value);
 }
}
```

Ans: -- The balance is: 500.0

167:

Objects communicate with each other by sending messages in the form of \_\_\_\_\_

class  
inheritance  
methods  
variables

ANS--- methods

168:

Consider the following code inside ABC.java file:

```
public class ABC{
 private static void main(String args[]){
 System.out.println("Hello World");
 }
}
```

What happens, when the program executes?

Compilation error

Runtime exception

Display "Hello World"

No output will be generated

Ans ---Runtime exception

=====

| Datatype | Default Value | Default size           |
|----------|---------------|------------------------|
| boolean  | false         | 1 bit (depends on JVM) |
| char     | '\u0000'      | 2 byte                 |
| byte     | 0             | 1 byte                 |
| short    | 0             | 2 byte                 |
| int      | 0             | 4 byte                 |
| long     | 0L            | 8 byte                 |
| float    | 0.0f          | 4 byte                 |
| double   | 0.0d          | 8 byte                 |

169:

Consider the following code inside ABCBank.java file

```
class ABCBank{
 int phone;
 double getPhone(){
 System.out.println("phone number received");
 return phone;
 }
}
```

What happens, when the program executes?

The code will not compile as there is no main method

The code will compile successfully and generate ABCBank.java file

The code will not compile as the return type in the getPhone method should be int not double

The code will compile successfully and generate ABCBank.class file

ANS-- The code will compile successfully and generate ABCBank.class file

170:

```
class ScopeOfVariables {
 int i = 34; // instance variable
 static int z; // class variable
 static {
 z = 10;
 System.out.println("inside static block: " + z);
 }
 void test() {
 int k = 200; // local variable
 System.out.println("local variable: " + (k + i));
 }
}
```

```
}

public static void main(String[] args) {
 ScopeOfVariables obj = new ScopeOfVariables();
 obj.test();
 System.out.println(obj.i);
}

}
```

ANS--inside static block: 10

local variable: 234

34

171: John is writing a class to handle the Account holder details. It requires two important variables as name and mobileNumber. For name, he has taken the datatype as String. For mobileNumber what should be the datatype?

int

float

long

char

ANS-- long

172: Which of the following datatype(s) can store 64-bit value?

int

char

byte

long

ANS-- long

173: What is the default value of a variable declared as boolean datatype?

0

1

true

false

ANS-- false

174: onsider the following code inside Account.java file:

```
class Account {
 double balance;

 public static void main(String args[]) {
 Account account1=null; //Line1
 Account account2=null; //Line2
 account1 = new Account(); // Line 3
 account2 = new Account(); //Line 4
 account2 = account1; //Line 5
 account1 = new Account(); // Line 6

 System.out.println(account1.balance);
 }
}
```

Which of the following statement is true after executing line 6?

ANS-- account1 and account2 are pointing to the different object

175: Consider the following code inside Bank.java file:

```
class Bank{
 String name;
 int createAccount(int accountNo,String accountHolderName){
 String name;
 int i=10;
 name= accountHolderName; // Line 5
 return i;
 }
}
```

In method createAccount at Line 5. What is the scope of variable “name”?

ANS-- local variable

176:

```
class Account {
 public static void main(String[] args) {
 double balance = 600;
 System.out.println("Amount to withdraw");
 double amount = 1500;
 if(amount < 0 || amount > balance) {
```

```
 System.out.println("Withdrawal has failed");
 }
 else {
 balance -= amount;
 System.out.println("Withdrawal has succeeded");
 }
}
```

Ans -- Amount to withdraw

Withdrawal has failed

177:

```
class Account {
 public static void main(String[] args) {
 double balance = 2000;
 System.out.println("Amount to withdraw");
 double amount = 500, limit = 10000, minbal = 500;
 if(amount <= limit && (balance - amount) > minbal) {
 balance -= amount;
 System.out.println("Withdrawal has succeeded");
 }
 }
}
```

ANS: Amount to withdraw

Withdrawal has succeeded

178:

Consider the following code inside ABC.java file

```
class ABC{

 public static void main(String args[]) {
 int x=100;
 int y=x;
 x++;
 int z=y;
 z++;
 System.out.println(y);
 }
}
```

What happens, when the program executes?

101  
102  
0  
100

ANS-- 100

179:

Consider the following code inside ABC.java file

```
class ABC{

 public static void main(String args[]){
 int m = 1;
```

```
int n = ++m + m++ + --m;
System.out.println(n);
}
}
```

What happens, when the program executes?

7  
6  
5  
8

ANS-- 6

180

Consider the following code inside JavaApplication6.java file

```
class JavaApplication6 {
 public static void main(String args[]){
 boolean a= true;
 boolean b=!true;
 boolean c= a|b;
 boolean d=a&b;
 boolean e= d?b:c;
 System.out.println(d+" "+e);
 }
}
```

What happens, when the program executes?

false  
true true  
true false  
false true

ANS-- false true

181:

What will be the output of the following code?

```
public class LogicalDemo {

 public static void main(String args[]) {
 boolean x = true;
 boolean y = false;

 System.out.println(!(x && y));
 System.out.println(x&&y);
 System.out.println(x| y);

 }
}
```

true false true

true true false

false false true

true true false

ANS-- true false true

182:

```
class Calculation{
 public static void main(String[] args){
 double d = 234.04;

 long l = (long)d; //explicit type casting

 int i = (int)l; //explicit type casting

 System.out.println("double value " + d);

 System.out.println("long value " + l);

 System.out.println("int value " + i);
 }
}
```

ANS---

double value 234.04

long value 234

int value 234

183:

```
class Calculation{
 public static void main (String[] args) {
 int i = 300;

 long l = i; //no explicit type casting

 float f = l; //no explicit type casting

 System.out.println("int value " + i);
 }
}
```

```
 System.out.println("long value " + l);
 System.out.println("float value " + f);
}
}
```

ANS --

```
int value 300
long value 300
float value 300.0
```

184:

Consider the following code inside ABC.java file.

```
class ABCBank{
 int phone=312345;
 double getPhone(){
 System.out.print("phone number received");
 return phone;
 }
 public static void main(String args[]) {
 ABCBank bank=new ABCBank();
 System.out.print(bank.getPhone());
 }
}
```

What happens, when the program executes?

```
display "phone number received"
display "phone number received 312345.0"
display "312345.0"
```

The code will not compile as getPhone should return double not int value

ANS ---display “phone number received 312345.0”

185:

What will be the output of the following code?

```
class TypeDemo{

 public static void main(String args[]){
 long x = 88843.78; // Line 1
 float y= x; //Line 2
 double z=x+y; //Line 3
 System.out.println(z);
 }
}
```

Compilation error occurs in Line 1

Compilation error occurs in Line 2

Compilation error occurs in Line 3

Executes successfully and display "177686.0"

ANS--- Compilation error occurs in Line 1

186:

Consider the following code:

```
class TypeDemo{
```

```
public static void main(String args[]){
 double price=30.0f;
 // other statements

}
```

Can a float value of 30.0f can be assigned to a double variable 'price' without type casting?

No

Yes

ANS-- YES

187:

Consider the following code:

```
class TypeDemo{

 public static void main(String args[]){
 byte b = 50; //Line 1
 b = b * 50;//Line 2
 System.out.println(b);
 }
}
```

What will happen after compilation?

Compilation error in Line 2 “b cannot contain value 100, limited by its range”

Compilation error in Line 2 “\* operator has converted b \* 50 into int, which cannot be converted to byte without casting”

Compilation error in Line 1 “b cannot contain value 50, limited by its range”

Compiles successfully without error

ANS-- Compilation error in Line 2 “\* operator has converted b \* 50 into int, which cannot be converted to byte without casting”

188:

Consider the following code:

```
class ReturnInt {
 returnType methodA(byte x, double y) //line2
 {
 return (short)x / y * 2;
 }
}
```

What is the narrowest valid returnType for methodA in line2?

int

byte

long

double

ANS--double

189:

```
class Account{
```

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

 double balance = 600;

 System.out.println("Amount to withdraw");

 double amount = 500;

 if (amount <= 0 || amount > balance){

 System.out.println("Withdrawal has failed");

 }else{

 balance-=amount;

 System.out.println("Withdrawal has succeeded");

 }

}
```

ANS-- Amount to withdraw

Withdrawal has succeeded

190:

```
class Account{

 public static void main(String[] args){

 double balance = 600;

 System.out.println("Amount to withdraw");

 double amount = 500;

 if(amount<=0){

 System.out.println("Withdrawal has failed as the amount is negative");

 }

 else if(amount>balance){

 System.out.println("Withdrawal has failed as the balance is low");

 }

 }

}
```

```
 else{
 balance-=amount;
 System.out.println("Withdrawal has succeeded");
 }
 }
}
```

ANS--Amount to withdraw

Withdrawal has succeeded

191:

```
class Account {
 public static void main(String[] args) {
 int choice = 0;
 choice = 2; // Input taken from user
 switch(choice) { // choice passed to switch statement
 // choice has to match the case value to execute the statements in that case option.
 case 1: // new record entry: newEntry()
 System.out.println("Entry deposition");
 break;

 case 2: // displaying details of account: display()
 System.out.println("Display operation");
 break; // break keyword is used to exit from the switch block.

 case 3: // deposit operation: deposit()
 System.out.println("Deposit operation");
 break;
 }
 }
}
```

```
default: // If no case matches, default will be executed.
 System.out.println("Invalid choice");
}
}
}
}
```

ANS -- Display operation

192:

```
import java.util.Scanner;

class Account {

 public static void main(String[] args) {

 double balance = 0, minbal = 500, depositAmt = 0;

 //Scanner sc = new Scanner(System.in); uncomment when working in eclipse

 do {

 System.out.println("$100 have been added to the account");
 depositAmt +=100; // harcode different depositAmt values
 //depositAmt = sc.nextInt(); uncomment when working in eclipse
 } while(depositAmt < minbal);

 balance = depositAmt;

 System.out.println("Transaction Complete");
 }
}
```

ANS-- \$100 have been added to the account

Transaction Complete

193:

```
import java.util.Scanner;

class Account {

 public static void main(String[] args) {
 double balance = 0, minbal = 500, depositAmt = 0;
 //Scanner sc = new Scanner(System.in); // uncomment when working in eclipse
 while(depositAmt < minbal) {
 System.out.println("$100 have been added to account");
 //depositAmt = sc.nextInt(); // harcode different depositAmt values
 //uncomment when working in eclipse
 depositAmt +=100;
 }
 balance = depositAmt;
 System.out.println("Transaction Complete");
 }
}
```

ANS--

```
$100 have been added to account
Transaction Complete
```

194:

```
class Account {
```

```
public static void main(String[] args) {
 double balance = 6000, rateOfInterest = 0.10, interest = 0;
 double withdrawal = 500, deposit = 600;
 for(int i = 1; i <= 12; ++i) {
 balance += deposit;
 balance -= withdrawal;
 interest = balance * rateOfInterest;
 balance += interest;
 System.out.println("The interest for the month " + i + " is " + interest);
 }
 System.out.println("The balance at the end of the year is " + balance);
}
}
```

ANS--

The interest for the month 1 is 600.0  
The interest for the month 2 is 660.0  
The interest for the month 3 is 726.0  
The interest for the month 4 is 798.6  
The interest for the month 5 is 878.46  
The interest for the month 6 is 966.3060000000002  
The interest for the month 7 is 1062.9366000000002  
The interest for the month 8 is 1169.2302600000003  
The interest for the month 9 is 1286.1532860000004  
The interest for the month 10 is 1414.7686146000005  
The interest for the month 11 is 1556.2454760600006  
The interest for the month 12 is 1711.8700236660006  
The balance at the end of the year is 18830.570260326007

195:

Bank application starts with a login screen. If the username="John" and password="Infy123", We need to display the message as "Welcome!" else we need to display "Sorry, wrong credentials!". What is the right way to do it?

if (username. equals("John") && password. equals("Infy123")) System.out.println ("Welcome!"); else System.out.println ("Sorry, wrong credentials!");

String s = (username. equals ("John") && password. equals("Infy123"))? "Welcome!": "Sorry, wrong credentials!"; System.out.println(s);

both option 1 and option 2

None of the option

ANS--both option 1 and option 2

196:

Bank application starts with a login screen. We need to display "welcome!" if the username and password are equal to "admin" and "Infy123". Else we want to display "Sorry, wrong credentials!". What is the right way to do it?

if (username. equals("admin") && password. equals("Infy123")) System.out.println ("welcome!"); else System.out.println ("Sorry, wrong credentials!");

If (username. equals("admin") || password. equals("Infy123")) System.out.println ("welcome!"); else System.out.println ("Sorry, wrong credentials!");

both option 1 and 2

None of the option

ANS-- if (username. equals("admin") && password. equals("Infy123")) System.out.println ("welcome!"); else System.out.println ("Sorry, wrong credentials!");

197:

Consider the following code:

```
class ABC{
 public static void main(String args[]){
 int i = 1,j = -1;
 switch (i) {
 case 0, 1:
 j = 1;
 case 2:
 j = 2;
 default:
 j = 0;
 }
 System.out.println("j="+j);
 }
}
```

What will happen on executing the code?

j = -1  
j = 0  
j = 1  
Compilation fails

ANS-- Compilation fails

198:

Consider the following code inside SwitchImpl.java file

```
class SwitchImpl {
```

```
public static void main(String[] args) {
 float f = 12;
 switch (f) { // Line 1
 case 10+1:
 System.out.println("Twelve"); // Line 2
 case 0:
 System.out.println("Zero"); //Line 3
 case (int)12.0:
 System.out.println("Decimal");
 default:
 System.out.println("Default");
 }

}
}
```

What will happen after compilation of the code?

Compiles successfully without error

Compilation fail in Line 2 “Cannot use operator in case”

Compilation fail in Line 3 “Cannot use zero in case”

Compilation fail in Line 1 “Cannot use float type in switch”

ANS--- Compilation fail in Line 1 “Cannot use float type in switch”

199:

Given:

```
int i = 1,j = 10;
do {
 if(i>j) {
 break;
 }
 j--;
} while (++i < 5);
System.out.println("i =" +i+ " and j =" +j);
```

What is the result?

i = 6 and j = 5  
i = 5 and j = 5  
i = 6 and j = 4  
i = 5 and j = 6  
i = 6 and j = 6

ANS--

200: i = 5 and j = 6

What will be the output of the underlying program?

```
class JavaApplication6 {
 public static void main(String args[]){
 int sum=0;
 for(int i=0,j=0;i<5&j<5;++i,j=i+1)
 sum+=i;
 System.out.println(sum);
 }
}
```

}

5

6

14

Compilation error

ANS-- 6

201:

Consider the following code snippet:

```
for (int i =0; i <3; i++) {
 switch(i) {
 case 0: break;
 case 1: System.out.print("one ");
 case 2: System.out.print("two ");
 case 3: System.out.print("three ");
 }
}
System.out.println("done");
```

What is the result?

done

one two done

one two three done

one two three two three done

ANS-- one two three two three done

202:

Consider the following code snippet:

```
class ForLoopDemo{

 public static void main(String args[]){
 for (int i=0; i < 10; i +=2) {
 System.out.print(i+"");
 ++i;
 }
 }
}
```

What will be the output of the code?

0369

02468

0248

369

ANS--

ANS -- 0369

203:

```
class Bank {
```

```
public static void main(String[] args) {
 int phone[] = new int[3]; // primitive type array
 phone[0] = 7120686;
 phone[1] = 7120687; // inserting element in an array
 phone[2] = 7120684;
 int count = 1;
 for(int i = 0; i < phone.length; ++i) { // traversing an array
 System.out.println("phone number " + count + ": " + phone[i]);
 ++count;
 }
}
}
```

ANS--

```
phone number 1: 7120686
phone number 2: 7120687
phone number 3: 7120684
```

204:

```
class bank{
 public static void main(String[] args){
 Customer customer[] = new Customer[2]; //Reference type Array
 Customer customer1 = new Customer("Anil", "Acc12345");
 Customer customer2 = new Customer("Ajay", "Acc12346");
 customer[0] = customer1; //storing in the array
 customer[1] = customer2;
 for(int i=0;i<customer.length;i++){ //traversing the array
 Customer customeObject = customer[i]; //retrieving customer Object
 String name = customeObject.displayCustomerName();
```

```

 System.out.println("the customer name is..." +name);

 }

}

}

class Customer{

 private String name;

 private String customerId;

 Customer(String uname, String ucustomerId){

 name = uname;

 customerId = ucustomerId;

 }

 public String displayCustomerName(){

 return name;

 }

}

```

ANS- the customer name is...Anil

the customer name is...Ajay

205:

Which of the following are valid array declarations/definitions?

- 1: int myArray1[5];
- 2: int myArray2[];
- 3: int myArray3[] = new int[5];

```
4: int myArray4[5] = new int[5];
5: int []myArray5 = new int[5];
6: int myArray6[] = new int[];
7: int myArray7[] = null;
```

1,2,3,6

2,3,5,7

2,4,5,7

3,4,5,7

ANS -- 2,3,5,7

206:

What is the output for the following program?

```
class ArrayProgram {
 public static void main(String args[])
 {
 int myArr[] = new int[] {0 , 1, 2, 3, 4, 5, 6, 7, 8, 9};
 int n = 6;
 n = myArr[myArr[n] / 2];
 System.out.println(myArr[n] / 2);
 }
}
```

0

1

3

6

ANS-- 1

207:

Which of the following statement is correct?

- i) int score[][]=new int[2][];
- ii) int score[][]=new int[2][2];
- iii) int score[][]=new int[][],3];
- iv) int score[][]=new int[2][];  
score[0]=new int[2];

i, ii, iii

i, ii, iv

ii, iii, iv

i, ii, iii, iv

ANS -- i, ii, iv

208:

```
class Acccount{
 public static void main(String[] args){
 double balance = 6000, rateOfInterest = 0.10, interest = 0;
 double withdrawal = 500, deposit = 600;
 int arr[] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12};
 for(int i: arr) { // The iteration in the loop happens automatically. The value is assigned to
```

```
//variable i from the array in every iteration of the loop.

balance += deposit; // Loop will repeat the statements in its body till the last element is reached
in the array.

balance -= withdrawal;

interest = balance * rateOfInterest;

balance += interest;

System.out.println("The interest for the month " + i + " is " + interest);

}

}

}
```

ANS--

```
The interest for the month 1 is 610.0
The interest for the month 2 is 681.0
The interest for the month 3 is 759.1
The interest for the month 4 is 845.0100000000001
The interest for the month 5 is 939.5110000000001
The interest for the month 6 is 1043.4621000000002
The interest for the month 7 is 1157.8083100000001
The interest for the month 8 is 1283.5891410000004
The interest for the month 9 is 1421.9480551000004
The interest for the month 10 is 1574.1428606100003
The interest for the month 11 is 1741.5571466710005
The interest for the month 12 is 1925.7128613381003
```

209:

Consider the following code snippet:

```
class LoopDemo{
 public static void main(String args[]){
 int arr[]={12,13,14,44};

 for(float i:arr){ // Line 1
 System.out.println(i);
 }
 }
}
```

What will be the output when this code snippet gets executed?

12 13 14 44

12.0 13.0 14.0 44.0

Compilation error in Line 1

ANS -- 12.0 13.0 14.0 44.0

210:

An online bookstore sells the following types of Books- Paper books and Electronic books. Both share a number of common attributes and functionalities, but also have a number of specific attributes and functionalities found only in each book distribution format.

Which feature of Object Oriented Concepts helps in designing these classes to increase the code reusability?

Abstraction

Encapsulation

Inheritance

Polymorphism.

## ANS-- Inheritance

211:

```
class Bank {
 private String bankName, area;
 private String phoneNumber;
 Bank() { // Default constructor
 bankName = "IBank";
 area = "Gandhi Nagar";
 phoneNumber = "9876543210";
 }
 Bank(String bname, String barea, String phoneNo) { // Parameterized constructor
 bankName = bname;
 area = barea;
 phoneNumber = phoneNo;
 }
 void displayBankDetails(){
 System.out.println("bank Name: " + bankName);
 System.out.println("Area of bank: " + area);
 System.out.println("Phone number of bank: " + phoneNumber);
 }

 public static void main(String[] args){
 Bank bank1 = new Bank(); //call default constructor
 Bank bank2 = new Bank("IBank", "Jaydev Nagar", "8876543219"); //call Parameterized constructor

 bank1.displayBankDetails();
 System.out.println("*****");
```

```
 bank2.displayBankDetails();
}
}
```

ANS -- bank Name: IBank

Area of bank: Gandhi Nagar

Phone number of bank: 9876543210

\*\*\*\*\*

bank Name: IBank

Area of bank: Jaydev Nagar

Phone number of bank: 8876543219

212:

Consider the following code inside Bank.java file:

```
class Bank {
 String bankName, area, phoneNo;

 public static void main(String args[]) {
 Bank bank = new Bank();
 System.out.println(bank.bankName);
 }
}
```

null

Compile time error as the constructor is not provided

Runtime exception as none of the variables are initialized

Executes successfully with no output

AND -- null

213:

Consider the following code inside Bank.java file:

```
class Bank {
 String bankName, area, phoneNo;

 bank(){
 System.out.println("Banking application");
 }

 public static void main(String args[]) {
 Bank bank = new Bank();
 System.out.println(bank.bankName);
 }
}
```

What happens, when the program executes?

Compile time error as the constructor name and class name are not same

Compile time error as the bank() method doesn't have return type specified

Print "Banking application" null

Print null

ANS-- Compile time error as the bank() method doesn't have return type specified

214:

Consider the following code inside Bank.java file:

```
class Bank{
 String bankName,area,phoneNo;
 void Bank(){
 System.out.println("inside constructor");
 }
 public static void main(String args[]){
 Bank bank=new Bank();
 System.out.println(bank.bankName);
 }
}
```

What will be the output when we run the code?

inside constructor null  
inside constructor  
null  
Runtime exception as none of the variables are initialized

ANS-- null

215:

Consider the following code inside Bank.java file:

```
class Bank{
 String bankName,area,phoneNo;
 public static void main(String args[]){
 Bank bank=new Bank();
 System.out.println(bank.bankName);
 Bank bank2=new Bank("abc","xyz","pqr");
 System.out.println(bank2.bankName);
 }
}
```

```
}

Bank(){

 System.out.println("inside constructor");

}

Bank(String x,String y,String z){

 bankName=x;

 area=y;

 phoneNo=z;

}

}
```

What will be the output when we run the code?

null abc  
print "abc" but with Runtime exception  
inside constructor null abc  
Runtime exception as none of the variables are initialized

ANS -- inside constructor null abc

216:

Consider the following code inside Bank.java file:

```
class Bank{

 String bankName;

 Bank(){

 bankName="TrustBank";

 }

}
```

Choose the correct option for the above code.

Not a valid Constructor declaration. As constructor is also a method it should have return type associated with it

Not a valid constructor declaration. As constructor should take some arguments

Code snippet is fine

Not a valid constructor. As it is having same name as class

ANS -- Code snippet is fine

217:

```
class Bank {
 private String bankName; //instance variable
 private String area;
 private String phoneNumber;

 Bank(String bankName, String area, String phoneNumber) { // Parameterized constructor

 this.bankName = bankName; //this keyword is used to assign
 this.area = area; //the value for instance variables
 this.phoneNumber = phoneNumber;

 }

 void displayBankDetails(){
 System.out.println("bank Name: " + bankName);
 System.out.println("Area of bank: " + area);
 System.out.println("Phone number of bank: " + phoneNumber);

 }
}
```

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

 Bank bank = new Bank("IBank", "Jaydev Nagar", "8876543210"); //call default constructor
 bank.displayBankDetails();

}

}
```

ANS--

bank Name: IBank  
Area of bank: Jaydev Nagar  
Phone number of bank: 8876543210

218:

What will be the output when we execute the following code snippet?

```
public class Bank{
 String bankName,area,phoneNo;
 public static void main(String args[]){
 Bank bank2=new Bank("abc","xyz","pqr");
 System.out.println(bank2.bankName);
 System.out.println(bank2.area);

 }
 Bank(String bankName,String area,String phoneNo){
 bankName=bankName;
 this.area=area;
 }
}
```

```
 }
 }
```

null xyz

Compilation error

abc xyz

Runtime Exception

ANS-- null xyz

219:

What would be the output when the following code is compiled/executed ?

```
class ABC{
 ABC (int i){
 System.out.println(i);
 this.i = i;
 }
 public static void main(String args[]){
 ABC obj = new ABC(10);
 }
}
```

0

10

null

Compile error "No variable i defined in class ABC

ANS-- Compile error "No variable i defined in class ABC

220:

What will be the output of the following code?

```
class ABC{
 int i;
 ABC(int i){
 System.out.println(i);
 this.i = i;
 }
 public static void main(String ka[]){
 ABC obj = new ABC(10);
 System.out.println(obj.i);
 }
}
```

10 10

Compilation error, instance variable “i” is not initialized

10 0

0 10

ANS-- 10 10

221:

```
class Loan {
```

```

int tenure;
double principal;
float interestRate;
String accountNumber;
public double calculateEMI(){
 double simpleInterest = (principal*interestRate*tenure)/100;
 double emi = (simpleInterest+principal)/tenure;
 return emi;
}
class HomeLoan extends Loan {
 HomeLoan() {
 tenure = 5; //reusing super class member variables
 principal = 20000;
 interestRate = 8.5f;
 accountNumber = "Acc12345";
 }
}

public static void main(String[] args) {
 HomeLoan hloan = new HomeLoan();
 double amount = hloan.calculateEMI(); // sub class Object
 // invoking super class method
 System.out.println("emi per year..." + amount);
}
}

```

ANS-- emi per year...5700.0

222:

What is the output of the following code snippet?

```
class Person{
 String name;
 Person(){}
 System.out.println("In Person class");
}

class Student extends Person{
 Student(){}
 System.out.println("In Student class");
}

int id;
public static void main(String a[]){
 Student ob=new Student(); //Line 1
 ob.name="abc"; //Line 2
 ob.id=123;
}
```

In Student class

In Person class In Student class

Compilation error in Line 2

In Student class In Person class

ANS-- In Person class In Student class

223:

Consider the following code inside Account.java file

```
class Person{
 String name;
 Person(){
 name="John";
 }
}

class Employee extends Person{
 int age;
 Employee(){
 age=34;
 }
}

class Customer extends Person{
 int salary;
 Customer(int salary){
 this.salary=salary;
 name="Maddy";
 }
}

public void displayDetails(){
 System.out.println(name+age+salary);
}
```

```
class Account {
 public static void main(String[] args) {
 Customer c=new Customer(20000);
 c.displayDetails();
 }
}
```

What will happen when the code is executed?

Code executes successfully and display “John3420000”

Code executes successfully and display “Maddy3420000”

Compilation fail “the variable age is not accessible from displayDetails method ”

Compilation fail “Person cannot have two child classes i.e. Employee and Customer”

ANS-- Compilation fail “the variable age is not accessible from displayDetails method ”

224:

```
class Loan{
 private float interest;

 Loan(){
 interest = 8.5f;
 }

 //calculateEMI overloaded methods
 public double calculateEMI(int tenure, double principal){
 double simpleInterest = (principal * interest * tenure) / 100;
 double emi = (simpleInterest+principal) / tenure;
 return emi;
 }
}
```

```

public double calculateEMI(double principal, int tenure){
 double simpleInterest = (principal * interest * tenure) / 100;
 double emi = (simpleInterest+principal) / tenure;
 return emi;
}

public double calculateEMI(int tenure, double principal, float interest){
 double simpleInterest = (principal * interest * tenure) / 100;
 double emi = (simpleInterest+principal) / tenure;
 return emi;
}

public static void main(String[] args){
 Loan loan = new Loan();
 double result = loan.calculateEMI(20000d, 5); //d means double
 double value = loan.calculateEMI(5, 20000d);
 double val = loan.calculateEMI(5, 20000, 9.5f); // f means float

 System.out.println("EMI per year is..." + result);
 System.out.println("EMI per year is..." + value);
 System.out.println("EMI per year is..." + val);

}
}

```

ANS---

EMI per year is...5700.0

EMI per year is...5700.0

EMI per year is...5900.0

225:

```
class Loan{
 public double calculateEMI(double principal){
 double simpleInterest = (principal * 8.5 * 5) / 100;
 double emi = (simpleInterest + principal) / 5;
 return emi;
 }
}
```

```
class HomeLoan extends Loan {
 // method overriden
 public double calculateEMI(double principal){
 int additionalTax = 200;
 double simpleInterest = (principal * 7.5 * 20) / 100;
 double emi = (simpleInterest + principal) / 20;
 return emi+additionalTax;
 }
}
```

```
class VehicleLoan extends Loan {
 // method overriden
 public double calculateEMI(double principal){
 int additionalTax = 200;
 double simpleInterest = (principal * 9.5 * 10) / 100;
 double emi = (simpleInterest + principal) / 10;
 return emi+additionalTax;
 }
}
```

```

}

class ExecuteLoan{

 public static void main(String[] args){

 Loan loan = null;

 loan = new HomeLoan(); //Runtime Polymorphism

 double hloan = loan.calculateEMI(2000000);

 loan = new VehicleLoan(); //sup class reference holding sub class Object

 double vloan = loan.calculateEMI(100000);

 System.out.println("Home loan emi per year is..." + hloan);

 System.out.println("Vehicle loan emi per year is..." + vloan);

 }

}

```

ANS--

Home loan emi per year is...250200.0  
 Vehicle loan emi per year is...19700.0

226:

What will the output of the below code snippet?

```

class OverloadingTest {

 public void display(String ref){

 System.out.println("String..."+ref);

 }
}

```

```
public void display(Object ref){
 System.out.println("Object..."+ref);
}

public static void main(String[] args) {
 OverloadingTest test=new OverloadingTest();
 test.display(null);

}
}
```

Object...null

String...null

Compilation error “both the methods are same”

Compilation error “cannot pass null as argument in display method”

ANS-- String...null

227:

Which of the following statements is/are correct in Java?

- (i) Constructor can be overloaded.
- (ii) Constructor can return a value.
- (iii) Constructor name should be the same as the class name.
- (iv) Constructor can take input parameters.

(i), (ii), (iii) and (iv)

(i), (iii) and (iv) only

(i) and (ii) only

(ii), (iii) and (iv) only

ANS-- (i), (iii) and (iv) only

228:

What is the output of the following code snippet?

```
class Person{
 String name;
 Person(){
 name="John";
 }

 public void getDetails(){
 System.out.println(name);
 }
}

class Employee extends Person{
 int age;
 Employee(){
 age=34;
 }

 public void getDetails(){
 System.out.println(name+age);
 }
}
```

```

class Customer extends Employee{
 int salary;
 Customer(int salary){
 this.salary=salary;
 name="Maddy";
 }

 public void getDetails(){
 System.out.println(name+age+salary);
 }
}

class Account {
 public static void main(String[] args) {
 Person c=new Customer(20000);
 c.getDetails();
 }
}

```

Maddy3420000

John3420000

John34

John

ANS-- Maddy3420000

229:

What is the output of the following code snippet?

```
class Person1 {
 String name;

 Person1() {
 System.out.println("In Person class");
 }

 void Print() {
 System.out.println("person name" + name);
 }
}

public class Student extends Person1 {
 void Print() {
 System.out.println("student name" + name);
 }

 Student() {
 System.out.println("In Student class");
 }

 int id;

 public static void main(String args[]) {
 Person1 person = null;
 person = new Student();
 person.name = "abc";
```

```
 person.Print();
}
}
```

In Person class In Student class student nameabc  
In Student class In Person class student nameabc  
Compilation error at person=new Student() line of code snippet  
Runtime exception at person=new Student() line of code snippet

ANS-- In Person class In Student class student nameabc

230:

```
class Loan{
 protected int tenure;
 protected float interestRate;

 Loan(int tenure, float interestRate){
 this.tenure = tenure;
 this.interestRate = interestRate;
 }
}

class HomeLoan extends Loan{
 HomeLoan(){
 super(5,8.5f); //invoking super class constructor
 }
 public double calculateEMI(double principal){
 double simpleInterest = (principal * interestRate * tenure) / 100;
```

```

 double emi = (simpleInterest + principal) / tenure;

 int additionalTax = 200;

 return emi + additionalTax;

 }

}

class ExecuteLoan{

 public static void main (String[] args) {

 HomeLoan loan = new HomeLoan(); //Runtime polymorphism

 double hloan = loan.calculateEMI(2000000);

 System.out.println("Home loan emi per year..." + hloan);

 }

}

```

ANS-- Home loan emi per year...570200.0

231:

```

class Loan {

 public double calculateEMI(double principal) {

 double simpleInterest = (principal*8.5*5) / 100;

 double emi = (simpleInterest+principal)/5;

 return emi;

 }

}

class HomeLoan extends Loan {

 public double calculateEMI(double principal) {

 int additiontax = 200;

 double emi = super.calculateEMI(principal); //calling super class method

```

```

 return emi + additionaltax;
 }

}

class ExecuteLoan {
 public static void main(String[] args) {
 Loan loan = null;
 loan = new HomeLoan(); // Runtime polymorphism
 double hloan = loan.calculateEMI(2000000);
 System.out.println("Home loan emi per year..." + hloan);
 }
}

```

ANS-- Home loan emi per year...570200.0

232:

What will be the output of the below code snippet?

```

class Loan{
 Loan(){
 System.out.println("Request for loan");
 }
}

class HomeLoan extends Loan{
 public static void main(String args[]){
 HomeLoan obh=new HomeLoan();
 }
}

```

Executes successfully, with no output displayed

Request for loan

compilation error as child class also should have a constructor, with super() in it

Runtime exception

ANS-- Request for loan

233:

What will be the output of the below code snippet?

```
class Loan{
 Loan(){
 System.out.println("Request for loan");
 }
}

class HomeLoan extends Loan{
 HomeLoan(){
 super();
 }

 public static void main(String args[]){
 HomeLoan homeLoan=new HomeLoan();
 }
}
```

Executes successfully, with no output displayed

Request for loan

Compilation error

Runtime exception

ANS-- Request for loan

234:

What will be the output of the following code snippet?

```
class Loan{
 Loan(int x){
 System.out.println("Request for loan");
 }
}
```

```
class HomeLoan extends Loan{
 HomeLoan(){
 System.out.println("Request for HomeLoan");
 }
 public static void main(String args[]){
 HomeLoan obh=new HomeLoan();
 }
}
```

Executes successfully, with no output displayed

Request for loan

Compilation error: Implicit super constructor Loan() is undefined. Another constructor must be invoked explicitly.

Runtime exception

ANS-- Compilation error: Implicit super constructor Loan() is undefined. Another constructor must be invoked explicitly.

235:

What will be the output of the following code snippet?

```
class Loan{
 int i;
 Loan(int x){
 i=x;
 System.out.println("Request for loan");
 }
}
```

```
class HomeLoan extends Loan{
 int i;
 HomeLoan(){
 super(10);
 i=super.i;
 System.out.println("Request for homeloan"+i);
 }
}
```

```
public static void main(String args[]){
 HomeLoan obh=new HomeLoan();
}
}
```

executes successfully, with no output displayed

Request for loan Request for homeloan 10

compilation error due to super.i statement

Runtime exception

ANS-- Request for loan Request for homeloan 10

236:

```
class Demo {
 final int tenure = 0;
 double principal;
 float interestRate;
 String accountNumber;
 final double calculateEMI(){
 return 2000;
 }
}
```

```
class Demo2 extends Demo{
```

```
// Error as final method is overriding
double calculateEMI(){
 return 8000;
}
```

```
}
```

```
class FinalDemo{
```

```
public static void main(String[] args) {
 Demo d = new Demo();
 d.tenure = 1; //Error as tenure is final
 System.out.println(d.tenure);
 System.out.println(d.calculateEMI());
}
}
```

ANS-- Compilation failed

file.java:14: error: calculateEMI() in Demo2 cannot override calculateEMI() in Demo

double calculateEMI(){

^

overridden method is final

file.java:23: error: cannot assign a value to final variable tenure

d.tenure = 1; //Error as tenure is final

^

2 errors

237:

final class Demo {

int tenure = 0;

double principal;

float interestRate;

String accountNumber;

double calculateEMI(){

return 2000;

}

}

```
class Dummy extends Demo{

 // Error as class is final
 double calculateEMI(){
 return 8000;
 }

}

class FinalDemo{

 public static void main(String[] args) {
 Demo d = new Demo();
 System.out.println(d.tenure);
 System.out.println(d.calculateEMI());
 }
}
```

ANS-- Compilation failed

file.java:11: error: cannot inherit from final Demo

```
class Dummy extends Demo{
^
1 error
```

238:

Which of the following is proper declaration of a final method in Java?

```
public final method();
final public void method();
public final void method(){ }
```

```
public void final method() { }
```

ANS-- public final void method(){ }

239:

If a variable is marked as final, it can be initialized

only once in the constructor

only once while declaring

only once either in the constructor or while declaring

any number of times

ANS-- only once either in the constructor or while declaring

240:

```
class Account{
```

```
 static int minbalance; //class variable
```

```
 static{
```

```
 minbalance = 500; // static block
```

```
}
```

```
 public static int getMinimumBalance(){
```

```
 return minbalance; //can't use instance variable in static method
```

```
 //and block
```

```
}
```

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

```
 System.out.println("The value.." + getMinimumBalance());
 }
}
ANS-- The value..500
```

241:

What will be the output of the following code snippet?

```
class Person {
 private String name;
 private static int age;
 static {
 name = "john"; // line 1
 }

 Person() {
 age = 20; // line 2
 }

 public int getAge() {
 return age; //line 3
 }
}

public class Account {
 public static void main(String[] args) {
 System.out.println(new Person().getAge());
 }
}
```

Compilation error in Line 1  
Compilation error in Line 2  
Compilation error in Line 3  
Code executes successfully and display 20

ANS-- Compilation error in Line 1

242:

What will be the output of the underlying program?

```
class AccessPoint{
 public int x1;
 static int y1;
 void calculates(int a, int b){
 x1+=a;
 y1+=b;
 }

}

public class JavaApplications6{
 public static void main(String[] args){
 AccessPoint obj1= new AccessPoint();
 AccessPoint obj2= new AccessPoint();
 obj1.x1=0;
 obj1.y1=0;
 obj1.calculates(1,2);
 obj2.x1=0;
```

```
 obj2.calculates(2,3);

 System.out.println(obj1.x1+" "+obj2.y1);

}

}
```

1 2

2 3

3 2

1 5

ANS-- 1 5

243:

What will be the output of the following code?

```
public class JavaApplication6{

 static int mfunc1()

 {

 System.out.println("Inside Func1");

 return 10;

 }

 static int mfunc2()

 {

 System.out.println("Inside Func2");

 }

 static int mfunc3(){

 System.out.println("Inside Func3");

 }

}
```

```
 return mfunc1()+mfunc2();
}

public static void main(String[] args){
 System.out.println(mfunc3());
}
}
```

Inside Func3 Inside Func1 Inside Func2 20  
Inside Func3 20  
Inside Func3 Inside Func1 Inside Func2 Inside Func1 20  
Compilation error

ANS-- Compilation error

244:

Which of the following is NOT a valid return type for a method in Java?

void  
int  
Integer  
static

ANS-- static

245:

```
class Employee{
 private String employeeId;
```

```
Employee(String employeeId){
 this.employeeId=employeeId;
}

public int reward(double...fixedDeposit){ //Variable argument
 double sum=0;
 int rewardPoint=0;

 for(double deposit:fixedDeposit){
 sum=sum+deposit;
 }

 if(sum>1000000){
 rewardPoint=20000;
 }

 else if(sum<1000000 && sum>=500000){
 rewardPoint=10000;
 }

 else{
 rewardPoint = 20000;
 }

 return rewardPoint;
}

public String getEmployeeId(){
 return employeeId;
}
}

}

class Execute{
 public static void main(String[] args){
 Employee employee1=new Employee("E1001");
 int rewardPoint=employee1.reward(100000,200000,300000);
 }
}
```

```
Employee employee2=new Employee("E1002");
int rewardPoint1=employee2.reward(100000,100000);
System.out.println(employee1.getEmployeeId() +" has got a reward of "+rewardPoint);
System.out.println(employee2.getEmployeeId() +" has got a reward of "+rewardPoint1);
}
}
```

ANS--

E1001 has got a reward of 10000

E1002 has got a reward of 20000

246:

Consider the following class inside Account.java file

```
public class Account {

void display(int accountId, int... amount){
 int sum=0;
 for(int x:amount){
 sum+=x;
 }
 System.out.println("The sum is: "+accountId+sum);
}

public static void main(String[] args) {
 Account account=new Account();
 account.display(1001,20,30,40,60,80);
}
```

}

What will be the output on executing the code?

The sum is: 1001230

The sum is:1231

Compilation fail “The variable argument type int of the method display must be the first parameter”

Compilation fail “the method display cannot have any other parameter if the variable argument exists”

ANS--

The sum is: 1001230

248:

```
enum Day{
 SUNDAY(1),MONDAY(2),TUESDAY(3),WEDNESDAY(4),THURSDAY(5),FRIDAY(6),SATURDAY(7);
 private int value;
 private Day(int value){
 this.value=value;
 }
 public int getValue(){
 return this.value;
 }
}
class UserInterface{
 public static void main (String[] args) {
 //printing all constants of an enum
 for(Day day:Day.values())
 System.out.println("Day:"+day.name()+" Value:"+day.getValue());
 }
}
```

ANS--

```
Day:SUNDAY Value:1
Day:MONDAY Value:2
Day:TUESDAY Value:3
Day:WEDNESDAY Value:4
Day:THURSDAY Value:5
Day:FRIDAY Value:6
Day:SATURDAY Value:7
```

249:

Consider the following code inside TestEnum.java file

```
enum Branches {

 Delhi(2),Chennai(3),Mumbai(3),Bangalore(5);

 private int noofOffice;

 private Branches(int noofOffice){
 this.noofOffice=noofOffice;
 }

 public int getNoofOffice(){
 return noofOffice;
 }
}

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

```
 System.out.println(Branches.Bangalore.getNoofOffice());
 }
}
```

What will happen after compilation of the code?

Compilation error “enum constructor cannot be private”

Compilation error “Cannot have variables in enum ”

Code will compile successfully

Compilation error “enum object should be created with new keyword”

ANS--

Code will compile successfully

250:

Consider the following class inside DelhiOffice.java file

```
abstract class Bank {
```

```
 private abstract void withdraw(); // Line 1
```

```
 abstract void deposit();
```

```
 public void balance(){} //Line 2
```

```
}
```

```
class DelhiOffice extends Bank{ // Line 3
```

```
 void deposit() { // Line 4
```

```
 // TODO Auto-generated method stub
```

```
}
```

```
}
```

The code will not compile due to

Compilation error in Line 1(abstract method cannot be private)

Compilation error in Line 2(abstract class cannot have concrete method)

Compilation error in Line 3(abstract class cannot be extended)

Compilation error in Line 4(deposit method should have public access modifier)

ANS--

Compilation error in Line 1(abstract method cannot be private)

251:

Consider the following class inside DelhiOffice.java file

```
abstract class Bank {

 private String bankName;

 Bank(String bankName) {
 this.bankName = bankName;
 }

 public String getBankName() {
 return bankName;
 }

}

class DelhiOffice extends Bank {

 DelhiOffice() {
```

```
 super("Axis Bank");

 }

public static void main(String[] args) {
 Bank bank = new DelhiOffice();
 System.out.println(bank.getBankName());
}

}
```

What will happen after compilation of the code?

Code will be compiled successfully

Compilation error will occur because “abstract class cannot have constructor”

Compilation error will occur because “abstract class must have an abstract method”

Compilation error will occur while invoking the super class constructor

ANS--

Code will be compiled successfully

Q 253:

```
interface IBankNew{
 boolean applyforCreditCard(Customer customer);
}
```

```
interface IBank extends IBankNew{
 int CAUTION_MONEY = 2000;
 String createAccount(Customer customer);
 double issueVehicleLoan(String vehicleType, Customer customer);
 double issueHouseLoan(Customer customer);
```

```
double issueGoldLoan(Customer customer);

}

class Customer {

 private String name;

 private String customerId;

 public String getName() {
 return name;
 }

 public void setName(String name) {
 this.name=name;
 }

 public String getCustomerId() {
 return customerId;
 }

 public void setCustomerId(String customerId) {
 this.customerId= customerId;
 }
}

class MumbaiBranch implements IBank {

 public String createAccount(Customer customer){
 return "Acc12345";
 }

 public double issueVehicleLoan(String vehicleType,Customer customer){
 if(vehicleType.equals("bike")) {
 return 100000;
 }
 }
}
```

```
 }

 else {
 return 500000;
 }
}

public double issueHouseLoan(Customer customer){
 return 200000;
}

public double issueGoldLoan(Customer customer){
 return 500000;
}

public boolean applyforCreditCard(Customer customer){
 return true;
}

}
```

```
class Execute{

 public static void main(String[] args){
 IBank bank=new MumbaiBranch();
 Customer customer = new Customer();
 customer.setCustomerId("cust1001");
 customer.setName("Ajay");
 String accountNumber = bank.createAccount(customer);
 System.out.println("Account number is..." +accountNumber);
 double gloan = bank.issueGoldLoan(customer);
 System.out.println("Gold loan amount is..." +gloan);
 double hloan = bank.issueHouseLoan(customer);
 System.out.println("House loan amount is..." +hloan);
 double vloan = bank.issueVehicleLoan("bike", customer);
 }
}
```

```
System.out.println("Vehicle loan amount is..." +vloan);
System.out.println("Caution money is..." +IBank.CAUTION_MONEY);
IBankNew bank1 = new MumbaiBranch();
boolean credit = bank1.applyforCreditCard(customer);
System.out.println("Credit card request.." + credit);
}
}
```

ANS--

```
Account number is...Acc12345
Gold loan amount is...500000.0
House loan amount is...200000.0
Vehicle loan amount is...100000.0
Caution money is...2000
Credit card request..true
```

Q 254:

```
interface IBank {
 int CAUTION_MONEY = 2000;
 String createAccount(Customer customer);
 double issueVehicleLoan(String vehicleType, Customer customer);
 double issueHouseLoan(Customer customer);
 double issueGoldLoan(Customer customer);
}

class Customer {
 private String name;
 private String customerId;

 public String getName() {
 return name;
 }
}
```

```
}

public void setName(String name) {
 this.name=name;
}

public String getCustomerId() {
 return customerId;
}

public void setCustomerId(String customerId) {
 this.customerId= customerId;
}

}

class MumbaiBranch implements IBank {

 public String createAccount(Customer customer){
 return "Acc12345";
 }

 public double issueVehicleLoan(String vehicleType,Customer customer){
 if(vehicleType.equals("bike")) {
 return 100000;
 }
 else {
 return 500000;
 }
 }

 public double issueHouseLoan(Customer customer){
 return 200000;
 }

 public double issueGoldLoan(Customer customer){
 return 500000;
 }
}
```

```

 }
}

class Execute{
 public static void main(String[] args){
 IBank bank=new MumbaiBranch();
 Customer customer = new Customer();
 customer.setCustomerId("cust1001");
 customer.setName("Ajay");
 String accountNumber = bank.createAccount(customer);
 System.out.println("Account number is..." +accountNumber);
 double gloan = bank.issueGoldLoan(customer);
 System.out.println("Gold loan amount is..." +gloan);
 double hloan = bank.issueHouseLoan(customer);
 System.out.println("House loan amount is..." +hloan);
 double vloan = bank.issueVehicleLoan("bike", customer);
 System.out.println("Vehicle loan amount is..." +vloan);
 System.out.println("Caution money is..." +IBank.CAUTION_MONEY);
 }
}

```

ANS--

Account number is...Acc12345  
 Gold loan amount is...500000.0  
 House loan amount is...200000.0  
 Vehicle loan amount is...100000.0  
 Caution money is...2000

Q 255:

Consider the following interface and class inside DelhiOffice.java file

```

interface IBank{
 void withdraw();
 void deposit();
}

abstract class Office implements IBank{

 public void withdraw() {
 System.out.println("Amount withdrawal");
 }
}

class DelhiOffice extends Office{
 public void deposit() {
 System.out.println("Amount deposit");
 }
}

public static void main(String[] args) {
 IBank bank=new DelhiOffice();
 bank.deposit();
}

```

What will happen after compilation of the code?

Compilation error “abstract class cannot implement an interface”

Compilation error “class Office should implement all the abstract methods”

Compilation error “cannot call deposit method on IBank reference”

Code will successfully compile

ANS--

Code will successfully compile

Q 256:

Which of the following holds true for classes and Interfaces? (Choose all that apply)

“Account extends SavingsAccount” is correct if and only if “Account is a class” and “SavingsAccount is an interface”

“Account extends SavingsAccount” is correct if and only if “Account is an interface” and “SavingsAccount is a class”

“Account extends SavingsAccount” is correct if “Account and SavingsAccount both are either interfaces or classes

“SavingsAccount implements Account” is correct if and only “Account is an interface” and “SavingsAccount is a class”

ANS--

“Account extends SavingsAccount” is correct if “Account and SavingsAccount both are either interfaces or classes

Q 257:

An \_\_\_\_\_ statement can be used to access the classes and interface of a different package from the current package.

instanceOf

import

implements

extends

ANS-- import

Q 258:

\_\_\_\_\_ class members can be accessed from:

classes that are present within the same package

Other classes that are subclasses of the declaring class and that are present in other packages

private

protected

default

Both option 1 and 3

ANS-- protected

Q 259:

You want the members of a certain class to be accessible only through public methods of the class.  
Which access modifier will accomplish this objective?

public

private

protected

transient

default access

ANS-- private

Q 260:

```
class Manager { // Outer Class
 private class Grade { // Inner Class
 private char grade;
 }
}
```

```
private char calculateGrade(String employeeid, int point) {
 if (isEmployeeExists(employeeid)) {
 if (point < 100 && point >= 90) {
 grade = 'A';
 } else if (point < 90 && point >= 80) {
 grade = 'B';
 } else {
 grade = 'C';
 }
 }
 return grade;
}

// Check if the employee id exists or not
private boolean isEmployeeExists(String employeeId) {
 // check from database or file system
 return true;
}

}
public char CheckEmployeeID(String employeeId, int point) {
 Grade grade = new Grade();
 return grade.calculateGrade(employeeId, point);
}

}
class Execute {
 public static void main (String[] args) {
 Manager manager = new Manager();
```

```

String employeeId = "l1001";
char gradePoint = manager.CheckEmployeeID(employeeId, 80);
System.out.println("The grade for " + employeeId + " is " + gradePoint);
}
}

```

ANS--

The grade for l1001 is B

Q 262:

```

interface Grade{
 char grades = 'A';
 public char calculateGrade(String employeeID, int point);
}

class Manager {
 Grade grade1 = new Grade(){} //Represents anonymous inner class
 private char grade;
 public char calculateGrade(String employeeID, int point){
 // calculate grade
 if (point < 100 && point >= 90) {
 grade = 'A';
 } else if (point < 90 && point >= 80) {
 grade = 'B';
 } else {
 grade = 'C';
 }
 return grade;
 }
};

public char checkEmployeeID(String employeeId, int point) {

```

```

 return grade1.calculateGrade(employeeId,point);

 }

}

class Execute {

 public static void main (String[] args) {

 Manager manager = new Manager();

 String employeeId = "l1001";

 char gradePoint = manager.checkEmployeeID(employeeId, 80);

 System.out.println("The grade for " + employeeId + " is " + gradePoint);

 }

}

```

ANS-- The grade for l1001 is B

Q 263:

Consider the following code inside Customer.java file

```

class Customer {

 private String name;

 private Customer(String name){

 this.name=name;

 }

 private class Account{ //Line 1

 private String accountNo="name is empty";

 int count=100;

 public String generateAccount(){

 if(!name.isEmpty()){//Line 2

```

```

 accountNo="Acc"+count+1;

 }

 return accountNo;
}

}

public static void main(String[] args) {

 Customer c=new Customer("John");//Line 3

 Customer.Account acc=c.new Account();

 System.out.println(acc.generateAccount());

}

}

```

What will happen after compilation of the code?

Compilation error in Line 1 “Inner class cannot be private”

Compilation error in Line 2 “Outer class variable can be used in Inner class through outer class instance only”

Code will compile successfully

Compilation error in Line 3 “cannot instantiate as outer class constructor is private”

ANS-- Code will compile successfully

Q 264:

What will be the output of the following code?

```

public class Account {

 public static void main(String args[]){
 Object[] o = new Object[3];//Line 1
 o[0] = 12;//Line 2
 int i = o[0]; //Line 3
 }
}

```

```
 System.out.print(i);

 }
}
```

Compilation error at Line 1 “cannot create Object type array”

Compilation error at Line 2 “cannot store integer value to an Object type array”

No compilation error, the code will be successfully executed and generate output 12

Compilation error at Line 3 “cannot convert from Object to int”

ANS-- Compilation error at Line 3 “cannot convert from Object to int”

Q 265:

What will be the output of the following code?

```
public class Account{
 public static void main(String argv[]){
 Integer iw = new Integer(2);
 Integer iw2 = new Integer(2);
 System.out.println(iw * iw2);
 System.out.println(iw.floatValue());
 }
}
```

Compile time error

Compilation and output of 4 followed by 2.0

Compilation and output of 4 followed by 2

Compile time error, the Integer class has no floatValue method

ANS-- Compilation and output of 4 followed by 2.0

Q 266:

Predict the output of the below code

```
package test;

public class Test
{
 public void compare(String shape)
 {
 if("null".equals(shape)) { // line 5
 System.out.println("Hi");
 }
 if(shape.equals("null")) { // line 8
 System.out.println("Hello");
 }
 }

 public static void main(String[] args) {
 Test t = new Test();
 t.compare(null);
 }
}
```

Hi Hello

NullPointerException in line 8

Compilation Error

NullPointerException in line 5

ANS-- NullPointerException in line 8

Q 267:

Consider the following code inside Bank.java file

```
class Bank {

 public void checkUsername(String name){

 String username="infosys";

 if(name.equals(username)){
 System.out.println("valid");
 }
 if(name==username){
 System.out.println("valid");
 }
 if(name.equals(new String("infosys"))){
 System.out.println("valid");
 }
 if(name==new String("infosys")){
 System.out.println("valid");
 }
 }

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

```
 new Bank().checkUsername("infosys");//Line 1
}
}
```

How many times "valid" will be printed?

2 times

3 times

4 times

Compilation error in Line 1 “Bank reference is missing”

ANS-- 3 times

Q 268:

Consider the following code inside Customer.java file

```
class Customer {

 public void getName(String fname,String lname){
 String name=fname.concat(lname);
 StringBuilder sb=new StringBuilder(fname);
 sb.append(" ");
 name=sb.toString(); //Line 1
 System.out.println(name);
 }

 public static void main(String[] args) {
 Customer c=new Customer();
 c.getName("sam","roy");
 }
}
```

}

What will happen on executing the main method of the code?

sam

samroy

sam roy

Runtime Exception in Line 1 “Cannot apply toString method on StringBuffer”

ANS-- sam

Q 269:

```
class AdvanceSalary {
 private LocalDate from;
 private LocalDate to;
```

```
 public AdvanceSalary(LocalDate from, LocalDate to) {
 this.from = from;
 this.to = to;
 }
```

```
 public void checkDates() {
 int compare = from.compareTo(to);
 System.out.println(compare);
 // Logic that makes sure from is not greater than to
 }
}
```

```
public class TechSol {
 public static void main(String[] args) {
 LocalDate from = LocalDate.of(2016, Month.JUNE, 01);
```

```
LocalDate to = LocalDate.of(2016, Month.JUNE, 03);
AdvanceSalary advance = new AdvanceSalary(from, to);
advance.checkDates();
}
}
```

What will be the value of compare when the above code is executed?

- 0
- 1
- 2
- 2

ANS-- (Ans-2)

Q 270:

```
public class TechSol {
 public static void main(String[] args) {
 Date startUtilDate = new Date();
 // Assume that the obtained date is Thu Nov 03 20:40:45 IST 2016
 Date tempUtilDate = startUtilDate;
 startUtilDate.setDate(26);
 System.out.println(tempUtilDate);

 LocalDate startLocalDate = LocalDate.of(2016, Month.JUNE, 01);
 LocalDate tempLocalDate = startLocalDate;
 startLocalDate = startLocalDate.plusDays(10);
 System.out.println(tempLocalDate);
 }
}
```

What will get consoled out during execution?

Sat Nov 26 20:40:45 IST 2016 2016-06-01

Sat Nov 03 20:40:45 IST 2016 2016-06-01

Sat Nov 26 20:40:45 IST 2016 2016-06-10

Sat Nov 03 20:40:45 IST 2016 2016-06-10

ANS--

Sat Nov 26 20:40:45 IST 2016 2016-06-01

Q 271:

```
public class DateTimeSample {
 public static void main(String[] args) {
 Period period = Period.of(1, 5, 2001);
 System.out.println(period.getDays() + "\t" + period.getMonths());
 }
}
```

What is expected to be consoled out?

1 5

5 1

2001 5

5 2001

ANS--2001 5

Q 272:

How many times may an object's finalize() method be invoked by the garbage collector?

Once

Twice

Not known

Zero times

ANS-- Once

Q273:

Consider the following code inside Account.java file:

```
class Account {

 double balance;

 public static void main(String args[]) {

 Account account1=null;

 Account account2=null;

 account1 = new Account();

 account2 = new Account();

 account2 = account1;

 account1 = new Account();

 Account account3=account1;

 account1=account3=null; // Line 8

 System.out.println("Welcome");
 }
}
```

How many instances are eligible for garbage collection in the above code after executing Line 8?

3

2

- 0
- 1

ANS- 2

Q274:

Which of the following statements is true for an immutable class?

This class doesn't have any setter methods to mutate the values of the private attributes.

All the attributes have private access specifier, restricting the object of the class to access the attributes.

This class can be extended.

Objects can be created using constructors provided in the class.

1 and 2

2 and 3

1 and 3

2 and 4

Ans - 1 and 2

Q275:

Which is the correct way of using for loop?

a.

```
for(int i=0; i<10; i++) {
 //Block of Code
}
```

b.

```
int i = 0;
for(i=0; i<10; i++) {
 //Block of Code
}
```

Only a is correct

Only b is correct

Both a and b are correct

Both a and b are not correct

ANS-- Only a is correct

Q276:

Which of the following is the valid reason for the class change under the single responsibility principle?

When the class has too many dependent objects.

When the constructor has one parameter.

When the method has one parameter.

All of these

ANS-- When the class has too many dependent objects.

Q277:

Which of the following statement is true about the lazy initialization of objects?

Objects of a class are created in bulk, and then used wherever required

Objects of a class are not created until they are required.

Due to large parameters dependency in constructor, the constructor invocation takes time and hence the time required to create objects is longer.

Objects are created using default constructor, and only required attribute values are initialized using setter methods in separate lines.

ANS-- Objects of a class are not created until they are required.

Q278:

Which of the following statement is correct about the memory leak in Java?

An unused object still being referenced causes memory leak.

Narrowing conversion of the variables causes memory leak.

Obsolete references cause memory leak.

All of the above

ANS-- All of the above

Q279:

```
import java.util.Calendar;
class CalendarDemo{

 public static void main(String[] args){

 Calendar cal=Calendar.getInstance(); //Create Calendar
 System.out.println("Current Date is\t:" + cal.getTime() +"\n"); //print current date
 cal.set(1992,0,1); //set the date to 1st Jan,1992
 System.out.println("Date after 1st modification\t:" + cal.getTime() +"\n");
 cal.add(Calendar.MONTH,3); //add 3 months
 System.out.println("Date after adding 3 months\t:" + cal.getTime() +"\n");
```

```
 cal.add(Calendar.YEAR,-3);//subtract 3 years
 System.out.println("Date after subtracting 3 years\t:" + cal.getTime() +"\n");

}
}
```

ANS--

Current Date is :Mon Aug 31 14:07:20 UTC 2020

Date after 1st modification :Wed Jan 01 14:07:20 UTC 1992

Date after adding 3 months :Wed Apr 01 14:07:20 UTC 1992

Date after subtracting 3 years :Sat Apr 01 14:07:20 UTC 1989

Q280:

We need to set the month of the calendar object to April. Which of the following options would apply to Line 1 of the code below?

```
import java.util.Calendar;

public class CalTester {

 public static void main(String[] args) {

 Calendar cal = Calendar.getInstance();
```

```
// Line 1

}

}

cal.set(Calendar.MONTH, 3);
cal.set(Calendar.MONTH, 4);
cal.set(Calendar.APRIL);
cal.set(Calendar.APRIL);

ANS-- cal.set(Calendar.MONTH, 3);
```

Q281:

```
class Student3{
 int id;
 String name;
 //method to display the value of id and name
 void display(){System.out.println(id+" "+name);}

 public static void main(String args[]){
 //creating objects
 Student3 s1=new Student3();
 Student3 s2=new Student3();
 //displaying values of the object
 s1.display();
 s2.display();
 }
}
```

ANS--

0 null

0 null

Q282:

```
class Student4{
 int id;
 String name;
 //creating a parameterized constructor
 Student4(int i,String n){
 id = i;
 name = n;
 }
 //method to display the values
 void display(){System.out.println(id+" "+name);}
```

```
public static void main(String args[]){
 //creating objects and passing values
 Student4 s1 = new Student4(111,"Karan");
 Student4 s2 = new Student4(222,"Aryan");
 //calling method to display the values of object
 s1.display();
 s2.display();
}
```

}

ANS--

111 Karan

222 Aryan

Q283:

```
class Student5{
 int id;
 String name;
 int age;
 //creating two arg constructor
 Student5(int i,String n){
 id = i;
 name = n;
 }
 //creating three arg constructor
 Student5(int i,String n,int a){
 id = i;
 name = n;
 age=a;
 }
 void display(){System.out.println(id+" "+name+" "+age);}

 public static void main(String args[]){
 Student5 s1 = new Student5(111,"Karan");
 Student5 s2 = new Student5(222,"Aryan",25);
 s1.display();
 s2.display();
 }
}
```

ANS--

111 Karan 0

222 Aryan 25

Q284:

```
class Student6{
 int id;
 String name;
 //constructor to initialize integer and string
 Student6(int i,String n){
 id = i;
 name = n;
 }
 //constructor to initialize another object
 Student6(Student6 s){
 id = s.id;
 name = s.name;
 }
 void display(){System.out.println(id+" "+name);}

 public static void main(String args[]){
 Student6 s1 = new Student6(111,"Karan");
 Student6 s2 = new Student6(s1);
 s1.display();
 s2.display();
 }
}
```

ANS--

111 Karan

111 Karan

Q285:

```
class Student7{
 int id;
 String name;
 Student7(int i, String n){
 id = i;
 name = n;
 }
 Student7(){
 }
 void display(){System.out.println(id+" "+name);}

 public static void main(String args[]){
 Student7 s1 = new Student7(111, "Karan");
 Student7 s2 = new Student7();
 s2.id=s1.id;
 s2.name=s1.name;
 s1.display();
 s2.display();
 }
}
```

ANS--

111 Karan

111 Karan

Q286:

```
class Student{
 int rollno; //instance variable
 String name;
```

```

static String college ="ITS";//static variable

//constructor

Student(int r, String n){

rollno = r;

name = n;

}

//method to display the values

void display (){System.out.println(rollno+" "+name+" "+college);}

}

//Test class to show the values of objects

public class TestStaticVariable1{

public static void main(String args[]){

Student s1 = new Student(111,"Karan");

Student s2 = new Student(222,"Aryan");

//we can change the college of all objects by the single line of code

//Student.college="BBDIT";

s1.display();

s2.display();

}

}

```

ANS--

111 Karan ITS  
222 Aryan ITS

Q287:

```

class Counter{

int count=0;//will get memory each time when the instance is created

```

```
Counter(){
 count++; //incrementing value
 System.out.println(count);
}

public static void main(String args[]){
 //Creating objects
 Counter c1=new Counter();
 Counter c2=new Counter();
 Counter c3=new Counter();
}
}
```

ANS--

```
1
1
1
```

Q288:

```
class Counter2{
 static int count=0; //will get memory only once and retain its value

 Counter2(){
 count++; //incrementing the value of static variable
 System.out.println(count);
 }

 public static void main(String args[]){
 //creating objects
```

```
Counter2 c1=new Counter2();
Counter2 c2=new Counter2();
Counter2 c3=new Counter2();
}
}
```

ANS--

```
1
2
3
```

Q289:

```
class Student{
 int rollno;
 String name;
 static String college = "ITS";
 //static method to change the value of static variable
 static void change(){
 college = "BBDIT";
 }
 //constructor to initialize the variable
 Student(int r, String n){
 rollno = r;
 name = n;
 }
 //method to display values
 void display(){System.out.println(rollno+" "+name+" "+college);}
}
//Test class to create and display the values of object
```

```
public class TestStaticMethod{
 public static void main(String args[]){
 Student.change(); //calling change method
 //creating objects
 Student s1 = new Student(111,"Karan");
 Student s2 = new Student(222,"Aryan");
 Student s3 = new Student(333,"Sonoo");
 //calling display method
 s1.display();
 s2.display();
 s3.display();
 }
}
```

ANS--

Output:111 Karan BBDIT

222 Aryan BBDIT

333 Sonoo BBDIT

Q290:

```
class A{
 int a=40; //non static

 public static void main(String args[]){
 System.out.println(a);
 }
}
```

ANS-- Output:Compile Time Error

Q291:

```
class Student{
 int rollno;
 String name;
 float fee;
 Student(int rollno,String name,float fee){
 rollno=rollno;
 name=name;
 fee=fee;
 }
 void display(){System.out.println(rollno+" "+name+" "+fee);}
}
class TestThis1{
 public static void main(String args[]){
 Student s1=new Student(111,"ankit",5000f);
 Student s2=new Student(112,"sumit",6000f);
 s1.display();
 s2.display();
 }
}
```

ANS---

0 null 0.0

0 null 0.0

Q292:

```
class Student{
 int rollno;
 String name;
```

```
float fee;

Student(int rollno,String name,float fee){

this.rollno=rollno;

this.name=name;

this.fee=fee;

}

void display(){System.out.println(rollno+" "+name+" "+fee);}

}

class TestThis2{

public static void main(String args[]){

Student s1=new Student(111,"ankit",5000f);

Student s2=new Student(112,"sumit",6000f);

s1.display();

s2.display();

}}
```

ANS---

111 ankit 5000

112 sumit 6000

Q293:

```
class Student{

int rollno;

String name;

float fee;

Student(int r,String n,float f){

rollno=r;

name=n;
```

```
fee=f;
}

void display(){System.out.println(rollno+" "+name+" "+fee);}
}
```

```
class TestThis3{

public static void main(String args[]){

Student s1=new Student(111,"ankit",5000f);

Student s2=new Student(112,"sumit",6000f);

s1.display();

s2.display();
}}
```

ANS---

```
111 ankit 5000
112 sumit 6000
```

Q294:

```
class A{

void m(){System.out.println("hello m");}

void n(){

System.out.println("hello n");

//m();//same as this.m()

this.m();
}
}

class TestThis4{

public static void main(String args[]){

A a=new A();
```

```
a.n();
}
}
```

ANS---

```
hello n
hello m
```

Q295:

```
class A{
A(){System.out.println("hello a");}
A(int x){
this();
System.out.println(x);
}
}

class TestThis5{
public static void main(String args[]){
A a=new A(10);
}}
```

ANS---

```
hello a
10
```

Q296:

```
class A{
A(){
this(5);
System.out.println("hello a");
```

```
}

A(int x){

System.out.println(x);

}

}

class TestThis6{

public static void main(String args[]){

A a=new A();

}}
```

ANS---

5  
hello a

Q297:

```
class Student{

int rollno;

String name,course;

float fee;

Student(int rollno,String name,String course){

this.rollno=rollno;

this.name=name;

this.course=course;

}

Student(int rollno,String name,String course,float fee){

this(rollno,name,course);//reusing constructor

this.fee=fee;

}

void display(){System.out.println(rollno+" "+name+" "+course+" "+fee);}
```

```
}
```

```
class TestThis7{
```

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

```
 Student s1=new Student(111,"ankit","java");
```

```
 Student s2=new Student(112,"sumit","java",6000f);
```

```
 s1.display();
```

```
 s2.display();
```

```
}
```

ANS---

111 ankit java null

112 sumit java 6000

Q298:

```
class Student{
```

```
 int rollno;
```

```
 String name,course;
```

```
 float fee;
```

```
 Student(int rollno,String name,String course){
```

```
 this.rollno=rollno;
```

```
 this.name=name;
```

```
 this.course=course;
```

```
 }
```

```
 Student(int rollno,String name,String course,float fee){
```

```
 this.fee=fee;
```

```
 this(rollno,name,course);//C.T.Error
```

```
 }
```

```
 void display(){System.out.println(rollno+" "+name+" "+course+" "+fee);}
```

```
}
```

```
class TestThis8{
 public static void main(String args[]){
 Student s1=new Student(111,"ankit","java");
 Student s2=new Student(112,"sumit","java",6000f);
 s1.display();
 s2.display();
 }
}
```

ANS---

Compile Time Error: Call to this must be first statement in constructor

Q299:

```
class S2{
 void m(S2 obj){
 System.out.println("method is invoked");
 }
 void p(){
 m(this);
 }
 public static void main(String args[]){
 S2 s1 = new S2();
 s1.p();
 }
}
```

ANS---

method is invoked

Q300:

```
class B{
 A4 obj;
 B(A4 obj){
 this.obj=obj;
 }
 void display(){
 System.out.println(obj.data);//using data member of A4 class
 }
}
```

```
class A4{
 int data=10;
 A4(){
 B b=new B(this);
 b.display();
 }
 public static void main(String args[]){
 A4 a=new A4();
 }
}
```

ANS---

Output:10

Q301:

```
class A5{
 void m(){
 System.out.println(this);//prints same reference ID
 }
}
```

```
public static void main(String args[]){
 A5 obj=new A5();
 System.out.println(obj);//prints the reference ID
 obj.m();
}
}
```

ANS---

A5@22b3ea59

A5@22b3ea59

Q302:

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

class Dog extends Animal{
 void bark(){System.out.println("barking...");}
}

class BabyDog extends Dog{
 void weep(){System.out.println("weeping...");}
}

class TestInheritance2{
 public static void main(String args[]){
 BabyDog d=new BabyDog();
 d.weep();
 d.bark();
 d.eat();
 }
}
```

ANS---

weeping...

barking...

eating...

Q303:

```
class Adder{
 static int add(int a,int b){return a+b;}
 static int add(int a,int b,int c){return a+b+c;}
}

class TestOverloading1{
 public static void main(String[] args){
 System.out.println(Adder.add(11,11));
 System.out.println(Adder.add(11,11,11));
 }
}
```

ANS---

22

33

Q304:

```
class Person{
 int id;
 String name;
 Person(int id,String name){
 this.id=id;
 this.name=name;
 }
}
```

```
}

class Emp extends Person{

float salary;

Emp(int id,String name,float salary){

super(id,name);//reusing parent constructor

this.salary=salary;

}

void display(){System.out.println(id+" "+name+" "+salary);}

}

class TestSuper5{

public static void main(String[] args){

Emp e1=new Emp(1,"ankit",45000f);

e1.display();

}}
```

ANS---

1 ankit 45000