**Interview Online Test**

2. OOPS

A constructor

Pick ONE option

* Must have the same name as the class it is declared within.
* Is used to create objects.
* May be declared private
* (a), (b) and (c) above.

Clear Selection

3. OOPS: What concepts come under Polymorphism in java?

Pick ONE option

* Method overloading
* Constructor overloading
* Method overriding
* All the above

Clear Selection

**4. Core Java: In java 8 Function is ?**

Pick ONE option

* Class
* Interface
* Lambda Expression
* Object

Clear Selection

**5. Core Java:What is Predicate in Java 8?**

Pick ONE option

* method
* class
* Interface
* Framework

**6. Core Java: Which method can be used to check null on an Optional variable in Java 8?**

Pick ONE option

* isPresent()
* isNullable()
* isPresentable()
* isNotNull()

**7. Core Java: Deque and Queue are derived from:**

Pick ONE option

* AbstractList
* Collection
* AbstractCollection
* List

**8. Core Java**

class A{

public void methodA()

{

System.out.println("MethodA");

}

}

class B extends A

{

private void methodA()

{

System.out.println("MethodB");

}

}

public class A1 {

public static void main(String[] s)

{

A a=new B();

a.methodA();

}

}

**what will be output of code snippet given above.**

Pick ONE option

* MethodA
* MethodB
* Compilation fails
* both a and b

**9. Core Java**

public class TestClass {

private String display (String t)

{

return "Hi "+t;

}

public static void main(String[] args) {

TestClass tc= new TestClassChild();

System.out.println(tc.display("Priyanka"));

}

}

class TestClassChild extends TestClass{

public String display (String t)

{

return "Hello "+t;

}

}

Pick ONE option

* Hi Priyanka
* Hello Priyanka
* CompilationError
* RuntimeError

**10. Core Java**

Below method abc is a legacy method and gets called from many places, for a new implementation, we need to introduce a new argument String c and use it for calculation. Select the best approach to achieve it.

public Class Util{

public static int abc(int a, String b)

{

??

10, 0000 lines of code..

?.

return result;

}

}

Pick ONE option

* Change the method directly by adding an argument and using it. Change all references so there are no compilation issues
* Overloading can help us achieve this but entire code will have to be copied to new method.
* Overloading without copying the code can help us achieve this
* Overriding the method will help us achieve this.

**11. Core Java**

public static void main(String[] args){

String str = "abc";

MyMethod(str);

System.out.println(str);

}

public static void MyMethod(String str)

{

Str = "def";

System.out.println(str);

}

Pick ONE option

* abc,abc
* abc,def
* def,def
* def, abc

Clear Selection

**12. Core Java**

**In what memory area variable temp and card written in main() get stored in ?**

Class CreditCard{

int num;

}

public class Bank{

public void static main(String [] args)

{

int temp;

CreditCard card;

}

}

Pick ONE option

* Heap, Heap
* Stack, Stack
* Heap, Stack
* Stack, Heap

**13. Core Java: What is the ouput of the program:**

static class MyClass{

int counter;

MyClass(int counter)

{if(counter(!=0)

{System.out.println("Creating Object with: "+ (counter-1) );

new MyClass(--counter);

this.counter = counter;

}

else

{ System.out.println("Stopping Object creation: "+counter);

}

}

}

public static void main(String [] args)

{

MyClass mc = new MyClass(5);

System.out.println(mc.counter);

}

}

Pick ONE option

5

0

4

None of the above

Clear Selection

14. Core Java

Find the output of the program:

public class Test{

public void static main(String [] srgs)

{ int a = 1;

int b = 2;

System.out.println(" ": +a+b);

}

}

Pick ONE option

* 3
* TRUE
* 12
* None of the above

Clear Selection

**15. Core Java**

**How many objects at line " Color red = Color.red" will be created?**

public class TestEnum{

enum Color{

red, blue, green;

}

public void static main(String [] args)

{

Color red = Color.red;

System.out.println("Color": +red.name());

}

}

Pick ONE option

* 1
* 2
* 3
* 0

**16. Hibernate:What will happen if we don?t have the non-args Contructor in the Entity bean?**

Pick ONE option

* Nothing will get impacted, as Class.newInstance() will automatically create no-args constructor
* HiberNateException will be thrown
* No Impact
* None of the above

**17. Hibernate: How many layers are there in Hibernate architecture?**

Pick ONE option

* **3**
* **4**
* **5**
* **2**

**18. Spring: Beans defined in Spring Framework are by default...?**

Pick ONE option

* Abstract
* Singleton
* Final
* Initialized

**19. Spring: Which of the following Java Collections types can you inject in the Spring Application?**

Pick ONE option

* Using List, Collections, maps or props tags
* Using List, set, maps or props tags
* Using List, set, maps or collection tags
* None of the above

**20. Spring: Element used to specify access attributes for bean’s methods?**

Pick ONE option

* security:protect
* security:intercept
* security:intercept-security
* None of the above

**21. Core Java: What will be the output ?**

public class HashMapTest{

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

˙˙˙˙˙˙˙˙˙ Map<Integer, String> map = new HashMap<Integer, String>();

˙˙˙˙˙˙˙˙˙˙map.put(120, "Uttart Pradesh");

˙˙˙˙˙˙˙˙˙˙map.put(11, "New Delhi");

˙˙˙˙˙˙˙˙˙˙map.put(null, "ABC");

˙˙˙˙˙˙˙˙˙˙map.put(null, null);

˙˙˙˙˙˙˙˙˙ System.out.println(map.size());

˙˙˙˙˙˙˙˙˙ System.out.println(map);

˙˙˙˙}

}

Pick ONE option

* NullPointerException
* {null=null, 120=Uttar Pradesh, 11=New Delhi}
* {null=ABC, null=null,120=Uttar Pradesh, 11=New Delhi}
* Compilation Error

**22. Core Java: What will be the output ?**

public class Test{

˙˙˙˙ public void myMethod(){

˙˙˙˙˙˙˙˙˙ private int i=0;

˙˙˙˙˙˙˙˙˙ System.out.println(i);

˙˙˙˙ }

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

˙˙˙˙˙˙˙˙˙ Test test = new Test();

˙˙˙˙˙˙˙˙˙˙˙test.myMethod();

˙˙˙˙ }

}

Pick ONE option

* 0
* Compilation error because declared i as private
* Compilation error because non-static method can't be called in static context
* None of the above

**23. Core Java: What will be the output ?**

public class JavaException{

public static void main(String[] args) {

˙˙˙˙˙˙˙˙˙ int˙d = 0;

˙˙˙˙˙˙˙˙˙ int˙n = 20;

˙˙˙˙˙˙˙˙˙ try{

˙˙˙˙˙˙˙˙˙˙˙˙˙˙˙int fraction = n/d;

˙˙˙˙˙˙˙˙˙˙˙˙˙˙ System.out.println("Value of fraction is "+fraction);

˙˙˙˙˙˙˙˙˙ } catch(Exception e){

˙˙˙˙˙˙˙˙˙˙˙˙˙˙˙ System.out.println(e);

˙˙˙˙˙˙˙˙˙˙˙˙˙˙˙ throw new ArithmeticException();

˙˙˙˙˙˙˙˙˙˙˙˙˙˙˙ System.out.println("Exiting from catch block");

˙˙˙˙˙˙˙˙˙ }

˙˙˙˙ }

}

Pick ONE option

* print on console : Java.lang.arithmetic exception
* : / zero
* Compilation error
* print on console : Java.lang.arithmetic exception
* : / zero
* Exiting from catch block
* print on console value of fraction

**24. Core Java: Which statement(s) is/are true about deserialization in java?**

Pick ONE option

* Parents classes of instance should be serialized
* If any super class in hierarchy is not serializable then it must have a default constructor
* JVM re-creates the serialized object without calling the constructor
* All the above

**25. Java: String Array: Which of the following Java declaration of the String array is correct?**

Pick ONE option

* String temp [] = new String {"j" "a" "z"};
* String temp [] = { "j " " b" "c"};
* String temp = {"a", "b", "c"};
* String temp [] = {"a", "b", "c"};

**26. Java : Threads: What is the output of the following Java snippet?**

class SampleDemo implements Runnable {

private Thread t;

private String threadName;

SampleDemo (String threadName){

this.threadName = threadName;

}

public void run()

{

while (true)

System.out.print(threadName);

}

public void start ()

{

if (t == null)

{

t = new Thread (this, threadName);

t.start ();

}

}

}

public class TestThread {

public static void main(String args[]) {

SampleDemo A = new SampleDemo( "A");

SampleDemo B = new SampleDemo( "B");

B.start();

A.start();

}

}

Pick ONE option

ABABABAB...(pattern repeats).

BABABABA...(pattern repeats).

AABAABAA...(pattern repeats).

A pattern cannot be predicted and can vary each time the program is run.

Clear Selection

27. Java : Collections

Which of the below statements are true about ArrayList and Vector in Java ?

Pick ONE option

Vector can be resized while ArrayList cannot be

Vector is synchronized while ArrayList is not

ArrayLists can grow but cannot shrink in size, while Vector can both grow and shrink

Vectors allow duplicate values while ArrayList doesnot

Clear Selection

28. Java: Synchronization

public class Test {

public synchronized void methodA(int i, String msg){

log.writeln(Integer.toString(i));

log.writeln(msg);

}

public void methodB(int i, String msg){

synchronized(this){

log.writeln(Integer.toString(i));

log.writeln(msg);

}

}

}

Consider the above code snippet, what of the following statements is true?

Pick ONE option

* synchronized keyword is applicable only to blocks and hence cannot be used for methodA.
* methodB is more efficient than methodA.
* Both the methods are equivalent.
* methodA is more efficient than methodB.

**29. Java : Threads: Which two statements are true?**

* Deadlock will not occur if wait()/notify() is used
* A thread will resume execution as soon as its sleep duration expires.
* Synchronization can prevent two objects from being accessed by the same thread.
* The wait() method is overloaded to accept a duration.
* The notify() method is overloaded to accept a duration.
* Both wait() and notify() must be called from a synchronized context.

Pick ONE option

* 1 and 2
* 3 and 5
* 4 and 6
* 1 and 3

**30. Which is the right answer to the following?**

**Which of the following is the root class (apex) of the exception heirarchy in Java?**

Pick ONE option

* Throwable
* Exception
* Error
* RuntimeException

**31. What is the expected result?**

import java.util.\*;

public class MyPancake implements Pancake {

public static void main(String[] args) {

List<String> x = new ArrayList<String>();

x.add("3"); x.add("7"); x.add("5");

List<String> y = new MyPancake().doStuff(x);

y.add("1");

System.out.println(x);

}

List<String> doStuff(List<String> z) {

z.add("9");

return z;

}

}

interface Pancake {

List<String> doStuff(List<String> s);

}

Pick ONE option

* [3, 7, 5]
* [3, 7, 5, 9]
* [3, 7, 5, 9, 1]
* Compilation fails.
* An exception is thrown at runtime.