**Java OOPS:**

1. What is OOPS?
2. Write basic concepts of OOPS?
3. What is a class?
4. What is an object?
5. What is Encapsulation?
6. What is Polymorphism?
7. What is Inheritance?
8. What is Method (operator) Overloading?
9. Why Method Overloading is not possible by changing the return type of method?
10. Draw Method Overloading and Type Promotion?
11. What is Method Overriding?
12. What are all the operators that cannot be overloaded?
13. What is the difference between Method Overloading and Method Overriding?
14. What are manipulators?
15. Define a constructor?
16. What is the difference between constructor and method?
17. What is Static Keyword?
18. What is this Keyword?
19. What is super Keyword?
20. What is Covariant Return Type?
21. What is Instance Initializer block?
22. What is final Keyword?
23. What is Runtime polymorphism?
24. What is an abstract class?
25. What is an interface?
26. Multiple inheritance is not supported in case of class but it is supported in case of interface, why?
27. What is marker or tagged interface?
28. What is Object Cloning in Java?
29. What is ternary operator?
30. What is finalize method?
31. What are different types of arguments?
32. What are tokens?
33. What are access modifiers?
34. What is the default access modifier in a class?
35. Whether static method can use non static members?
36. What is static and dynamic binding?
37. What is Instanceof Operator?
38. How many instances can be created for an abstract class?
39. Which OOPS concept is used as reuse mechanism?
40. Which OOPS concept exposes only necessary information to the calling functions?

**Core Java:**

1. What do you understand by JVM?
2. Explain Internal Architecture of JVM.
3. What is JRE?
4. What do you understand by JDK?
5. What is JIT (Just-In-Time) compiler?
6. What is platform?
7. What is the main difference between Java platform and other platforms?
8. What gives Java its ‘write once and run anywhere’ nature?
9. What is classloader?
10. Is Empty .java file naming a valid source file name?

**String Handling:**

1. What is String? Is it type of data type or not?
2. Why String is an immutable Object?
3. What is difference between “String” and “new String()” in java?
4. How to Split String in java?
5. How to compare string?
6. What is String Pool?
7. How to concat two different String?
8. What is StringTokenizer Class?
9. How trim() works in string?
10. How will you sort a collection of String in case sensitive order?
11. How does substring() method works?
12. Difference between length and length()?
13. What is StringBuffer class and what are the methods in StringBuffer Class?
14. What is StringBuffer class and what are the methods in this class?
15. Difference between String, StringBuffer and StringBuilder?
16. How to create Immutable class?
17. What is the purpose of toString() method in java?

**Exception handling:**

1. What is an Exception? Explain the exception hierarchy in java?
2. How are the exceptions handled in java?
3. Why exceptions are handled?
4. What is difference between Error and Exception?
5. What is Runtime Exception or unchecked exception?
6. What is checked exception?
7. What is difference between ClassNotFoundException and NoClassDefFoundError?
8. What is throw Keyword?
9. What is use of throws Keyword?
10. What is difference between throw and throws?
11. What are the possible combination to write try, catch , finally block?
12. How to create custom Exception?
13. When to make a custom checked Exception or custom unchecked Exception?
14. What is Stack Overflow Error?
15. Why did the designers decide to force a method to specify all uncaught checked exceptions that can be thrown within its scope?
16. Once the control switches to the catch block does it return back to the try block to execute the balance code?
17. Where is the clean-up code like release of resources is put in try-catch-finally block and why?
18. Is it valid to have a try block without catch or finally?
19. Is it valid to place some code in between try the catch/finally block that follows it?
20. What happens if the exception is never caught and throws down the method stack?
21. How do you get the descriptive information about the exception occurred during the program execution?
22. Can you catch more than one exception in a single catch block?
23. Why it is not considered as a good practice to write a single catchall handler to catch all the exceptions?
24. What is exception matching?
25. What happens if the handlers for the most specific exceptions are placed above the more general exceptions handler?
26. Does the order of the catch blocks matter if the Exceptions caught by them are not subtype of each other?
27. What happens if a method does not throw an checked Exception directly but calls a method that does? What does ‘Ducking’ the exception mean?
28. Is an empty catch block legal?
29. Can a catch block throw the exception caught by itself?