```
1. Which version of java are you using ?
4. Can you override static method in java ?
5. Why java does not support operator overloading ?
6. Why String is immutable in java ?
7. Is String is thread safe in java ?
8. What is string tokenizer ?
```

2. How to create thread-safe singleton class in java using double checked locking ?

```java
// Lazy Initialization Using double check locking
public class SingletonClass {

    private static volatile SingletonClass instance = null;

    private SingletonClass() {

    }

    // Double Checked Locking Pattern
    public static SingletonClass getInstance() {
        if(instance == null) {
            synchronized (SingletonClass.class) {
                if(instance == null) {
                    instance = new SingletonClass();
                }
            }
        }
        return instance;
    }
}
```

3. Why multiple inheritance is not supported in java ?

```
In Java, Multiple Inheritance is not supported
through Class but it is possible by Interface, why?
• As we have explained in the inheritance chapter, in multiple
inheritance, subclasses are derived from multiple superclasses.
• If two superclasses have the same method name then which method
is inherited into subclass is the main confusion in multiple
inheritance.
• That’s why Java does not support multiple inheritance in case of class.
But, it is supported through an interface because there is no
confusion. This is because its implementation is provided by the
implementation class.
```

9. What is String constant pool in java ?

- String constant pool is a special memory area in heap which is used for storing string objects. Internally, the string class uses a string constant pool.

```
10. Write a program to reverse a string ?
```

11. What are different ways to create a string object ?

- Generally, there are two ways to create the string object in java.
- They are:
    - By string literal
    - By new keyword

```
//String literal: String literal is created by using double quotes. For
example:
String s = "Hello";

// New keyword: It is the second way of creating string object in java.
// It is just like creating an object of any class. We can declare it as:
String s = new String("Hello");
```

```
12. List of diff strings, how to concat all string and create a new string
?
13. Why char array is preferred to store password than string in java ?
14. What is difference between Stringbuilder and Stringbuffer in java ?
15. What happens if your Serializable class contains a member which is not
serializable? How do you fix it ?
- https://www.youtube.com/shorts/IpwvQlyyEDY
16. How to create immutable class in java ?
```

17. What is use of intern() method in java ?

```
public class InternExample {
    public static void main(String[] args) {
        String s1 = new String("hello");
        String s2 = "hello";

        // Before interning
        System.out.println(s1 == s2); // false, different references

        // Interning s1
        s1 = s1.intern();
```

```
        // After interning
        System.out.println(s1 == s2); // true, same reference
    }
}
```

18. Write a program to remove duplicates from any array ?
19. How do you break a singleton pattern ?
20. How do prevent a singleton pattern from breaking ?
21. What are new features of java 8 ?
22. What is Lambda expression in Java 8 ?
23. What is metaspace in java 8 ?
24. What is use of functional interface in Java 8 ?
25. In which scenario you will use parallel stream in java 8 ?
26. Have you used parallel streams as well ?
27. How does lambda expression relates to functional interfaces ?
28. Can you tell me 3 terminal operators used in streams ?
29. What are the other features of Java 8 you have used while coding ?
30. Can you iterate and remove the element of ArrayList at the same time ?
31. what was log4j vulnerability?, how to fix log4j vulnerability ?
32. Can you tell me how hashmap internally works ?
33. What is concurrent hashmap ?
34. What is weak hashmap ?
35. How load factor works in hashmap ?
36. Difference between Hashmap vs Hashtable ?
37. What will happen if we put key object in hashmap which is already there ?
38. Difference between Stack and Queue ?
39. What is Hashtable ?
40. How blocking queue works ?
41. Difference between hashmap and concurrent Hashmap ?
42. Lets say you are working with hashmap, First key is null, value is Ravi, Second key is null, value is Bisht, hashmap.get(null), what will I get
43. Explain the FailFast iterator and FailSafe iterator along with examples for each
44. Explain lamda expression and functional expressions ?
45. Why we are using functional annotation for functional interface
46. Lets say you have two variable a and b, with values 5 and 10. Interchange value of a and b without using a temporary variable ? Will the program work for negative numbers ?
47. If catch is not present, only finally is present, will the code work ?
48. multiple catch with single finally will work ?
49. If insert can happen by PUT why do we have POST
50. can we fetch data with the help of POST ?

51. What is functional interface in java ?
52. Explain some of the functional interfaces which are used in java ?

53. How to make class threadsafe ?
54. What is Eureka Server
55. How do u create custom exceptions ?
56. When will you get 415 & 504 http response code ?
57. Email field has not null contrainst, how to remove not null constraint from not null column
58. What is difference between new Arraylist and Arrays.asList
59. Which one is fast LinkedList or Arraylist ?
60. Which case would you prefer LinkedList and which case Arraylist ?
61. Difference between Aggregation and composition
62. What is Fault tolerance
63. Jwt authentication related questions
64. Heap Memory and JVM
65. How do you create HTTP POST request in Java
66. What is the use of super keyword
67. What is throw, throws and when do we use these ?
68. What is synchronization ?
69. What is a treeset ?
70. Explain the role of API Gateway in a microservices architecture ?
71. Explain functional interface and what are some of the available interface ?
72. What are REST API Best practices ?
73. What is an one to one mapping and how to set up it with JPA/Hibernate ?
74. Difference between entity object and value object ?
75. What is the difference between ClassNotFoundException and NoClassDefFoundError
76. is finally block always be executed
77. What is an exception in java and how you handle exception in java ?
78. Explain classloader in java ?
79. Can we override static method in java ?
80. What is immutable classes & how we can create a new immutable class ?
81. Difference between comparator and comparable in java ?
82. What is hash collision ?
83. How will you handle Null pointer exception ?
84. What is transient in serialization ?
85. Can we serialize static variables ?
86. What is volatile keyword ?
87. What are design patterns you have worked on ?
88. How to create a singleton class ?
89. What is garbage collection ?
90. There is a method called GC, what is the purpose ?
91. How garbage collection works internally in java ?
92. What is use of actuators ?
93. Diff between constructor and setter injection ?
94. How we can read the property from property file ?
95. What is use of @ Value annotation ?
96. What is CI and CD, and why it is important ?
97. Why the main method is static in java ?
98. Can we override private variable in java ?
99. What is blank final variable in java, If we declare a final variable but we havenot assigned any value, what happens here ?
100. How the java objects are stored in memory ?

101. List of mobile numbers, find customers having number +91 ?
102. What is Predicate in terms of functional interface ?
103. Explain builder design pattern ?
104. How security is managed by your application ?
105. Do you know what is JWT2.0 and JWT token ?
106. Advantages of hibernate over JDBC ?
107. What is session in hibernate and is it thread safe ?
108. What is lazy loading in hibernate ?
109. can you override private and static method ?
110. What happens if an exception is thrown by Super class ?
111. What is wrapper class ?
112. What is garbage collection ?
113. Write a singleton design pattern in multi threaded environment ?
114. You need to store password .What type of datatype you will use?
115. What is the difference between factory and abstract factory design pattern ?
116. Difference between sleep() and wait();
117. Can you access non static variable in static contest ?
118. Suppose you declare variables outside main method . how to access variables in main method?
119. What is the difference between Comparator and Comparable in java
120. Is it better to make method synchronized or critical section synchronized?
121. in try catch and finally we return the value, priority is in which block ?
122. What is a Aggregation ?
123. Have you used Hibernate. What is the difference between hibernate first level cache and second level cache.
124. What is Inversion Of Control ?
125. How garbage collection works in java ?
126. Which scenario have you used Optional class & how optional class is really helpful ?
127. Which scenario we will use arraylist and when we will use linked list ?
128. Do you know what is Singleton ?
129. what are objectives of get put post and delete methods ?
130. API, Can we use get method to create resourses ?
131. Employee Service class will be annotated with which annotation ?
132. What would happen if I dont import java.lang method ?
133. Have you used serialization anywhere in your application ?
134. What is response entity and benefit of using response entity ?
135. What is docker and docker image and docker container ?
136. Suppose I have list and I want to reverse it, how can i do that ?
137. Can you tell me real world example of encapsulation ?
138. What is stateless object ?
139. Can you tell me difference @ Component @ Service and @ Repository annotation ?
140. Reason of overriding equals and hashcode method ?
141. Have you used reflections anywhere ?
142. How can we create object dynamically at runtime in java ?
143. Is it possible to rethrow an exception ?
144. Hashset vs TreeSet, which one is better in which scenario ?

145. What is use of Generics in java ?
146. What is use of CopyonWrite Arraylist ?
147. Synchronized block or method as synchronized, which one is better ?
148. What type of IDE you are using ?
149. Can you tell me the difference between truncate and delete ?
150. What are the different types of exceptions you have faced till now
151. What is order of exception in java, which is highest
152. Where will be a new object stored- heap or stack?
153. What is instance level locking and class level locking ?
154. Difference betwen normal interface and functional interface ?
155. There is primitive data type int and wrapper class Integer. Which will take more memory ?
156. I am expecting an exception in one of my code, how can i write test case for that ?
157. Difference between unit test and integration test ?
158. What is test driven development and what do you think about it ?
159. Database ACID properties
160. What are joins in database ?
161. In database, when we use union, intersection. suppose same thing we want to apply in JPA ?
162. How are you saving data into the database
163. Prepared Statements vs Statement in JDBC
164. SQL queries with group by
165. Primary Key Vs Unique Key in database ?
166. What are triggers in database ?
167. SQL query to fetch the name of employee who earn the highest salary ?

168. There is a table called employee in the database, id fn ln , there are few employee records. Now I decided to remove a column phone number. How can i delete it from the table?
169. What is callable interface in threads ?
170. In threads there is a sleep method and there is a wait method, what is the difference ?
171. I want to suspend a thread for a time of 5 seconds, how to do this ?
172. In thread there is shutdown hook, do you know what it is ?
173. How can you make a piece of code thread safe ?
174. How can you take a thread dump in java
175. What is difference between thread and process ?
176. What is @Autowire ?
177. What are spring boot components ?
178. Why spring boot was introduced ?
179. Advantages of using spring boot applications ?
180. How auto configuration works in spring boot ?
181. Do you know why we use @qualifier annotation ?
182. What is bean wiring in spring ?
183. How can I create a spring rest application from scratch ?
184. Which scnearios we will use @component vs @bean in spring boot
185. What @SpringBootApplication will do ?
186. What are spring profiles ?
187. What is actuator in spring boot ?
188. Do you know how to develop an api in spring

```
189. Explain the purpose of Autowired Annotation in spring ?
190. Main difference between spring and spring boot ?
191. What are scopes of spring bean ?
192. Difference between SpringBoot and Spring Framework ?
193. What is Spring AOP
194. Give examples of commonly used spring boot starters ?
195. @Controller vs @RestController in Spring
196. How a Spring boot application bootstraps ?
197. Which methods are there in comparable ?
198. Difference between compile time and run time exception ?
199. What is static initializer block ?
200. Where the static blocks gets stored inside memory ?
201. What is Externalization in java ?
202. What is use of generics in java ?
203. How to disable a specific auto-configuration class ?
204. What are actuator-provided endpoints used for monitoring the spring
boot application ?
205. Can we create a custom annotation ?
206. When we extend the thread class what all method we override and where
thread starts ? What is the case when they implement runnable ? any
difference ? Can we call start method 2 times
207. What is exception order in catch block ?
208. What all collections have you used in your project ?
209. What is equals and hashcode contract ?
210. Do you know hierarchy in exceptions ?
211. What is classpath ?
212. Difference between jdk, jre and jvm ?
213. How does jvmcontributes to java's platform independence ?
214. Describe the role of RestController and RequestMapping annotation in
spring ?
215. How does spring data JPA work with hibernate for ORM ?
216. In springboot application how would you optimize the data access ?
217.  Why spring buffer is called mutable ?
218. Lets say you have developed a rest api, REST api is not able to get
that particular resource. What will you return to the client ?
219. What is use of an interface
220. Difference between static and non-static method ?
221. Difference between array and arraylist ?

... will add more real interview questions, check weekly ...
...
Visit my website for focussed learning
https://www.myjavainterview.in/core-java
```

```
1. What are conditional annotations in spring boot ?
2. What is the default server provided by spring boot ?
3. How does spring boot support the development of restful web services ?
4. Explain role of EnableAutoConfiguration in spring boot app and how does
spring achive auto configuration internally ?
5. Are you aware of actuator endpoints?
6. What is dependency name for actuator ?
```

```
7. Can we secure these actuator endpoints ? and how can we secure it ?
8. Wht strategies would be used to optimize the performance of spring boot
applications ?
9. What is Aspect Oriented Programming ?
10. Can you give a usecase of AOP ?
11. Can you explain how spring boot profile ?
12. Which file do you use yml or properties in your current project ?
13. Do we have any benefit of using yml over properties file ?
14. Can you explain component scan and how would you prevent a specific
package from being scanned ?
15. Can you explain what is default embedded server provided by the spring
boot ?
16. Can we change default embedded server ?
17. How does AOP differs from the traditional programming models ?
18. How to create global exception handler in spring boot application ?
19. How will you call a restful web service from a spring boot application
?
```

20. What is application context and bean factory in spring boot ?

- BeanFactory is the basic container whereas ApplicationContext is the advanced container.
- ApplicationContext extends the BeanFactory interface.
- ApplicationContext provides more facilities than BeanFactory such as integration with spring AOP, message resource handling for i18n etc.

```
21. What is inversion of control(IOC) or dependency Injection(DI) ?
22. Do you know different types of injections ?
23. How do you handle transactions in spring boot application ?
24. Why we should use spring boot ?
25. What annotations you have used in your spring boot application ?
26. PostMapping vs PutMapping ?

– check weekly,,, will add more
```

Why we are using JPA and for database interaction we can use JDBC also.

Internally JPA How it is work.

```

```

What are Object class methods?

```
getClass()
hashCode()
wait()
notify()
notifyAll()
```

```
toString()
clone()
equals()
finalize()
```

What is HashCode() in Java ?

```
- for Every object jvm will generate a unique number which is nothing but
hashCode.
- JVM using hashCode while saving objects into hashing related data
structures like HashSet, HashMap, and HashTable etc.
```

Various ways to create object ?

```
- Using new Keyword
- Using clone() method
- Using Deserialization
- Using newInstance() method of the Class class
- Using newInstance() method of the Constructor class
```

@Transient Annotation

```
@Getter
@Setter
@AllArgsConstructor
@NoArgsConstructor

@Entity
@Table(name = "employees")
public class Employee{

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private long id; //bigint

    private String name; //varchar(255)

    // Specifies that the property or field is not persistent.
    //It is used to annotate a property or field of an entity class,
mapped superclass, or embeddable class.
    @Transient
    private String gender;
}
```

@Component Annotation

- The @Component annotation indicates that an annotated class is a "Spring Bean/Component".
- The @Component annotattion tells Spring container to automatically create Spring bean.
- responsibility of creating bean is given to springboot
- It is a class-level annotation.
- It is used to mark a Java class as a bean.
- A Java class annotated with @Component is found during the classpath .
- The Spring Framework pick it up and configure it in the application context as a Spring Bean.

```
- Cloning in Java
https://www.youtube.com/watch?v=e1Mw-vNXuFY
```

```
Filter and Find Even Number using java 8
Find Sum of Squares using java 8
Sort a list of string using java 8
Double Each Number of List using java 8
Find distinct element from the list using java 8


https://www.youtube.com/watch?v=BESfZAZnABs&t=3s
```

How to sort a Map based on keys or values using stream?

```
public class MapExample {

    public static void main(String[] args) {


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

        map.put("ABHAY KASHINATH PATIL", 416404);
        map.put("ANUP SANJAY DHOTRE", 457030);
        map.put("ADV NAJIB SHAIKH", 3300);
        map.put("NOTA", 5783);
        map.put("AMBEDKAR PRAKASH YASHWANT", 276747);

        for(Map.Entry<String,Integer> m:map.entrySet()) {
            System.out.println(m.getKey()+"  " + m.getValue());
        }

        //How to sort a Map based on keys or values using stream
```

```
        TreeMap<String, Integer> tm = new TreeMap<String, Integer>(map);

        System.out.println(tm);

        //Sort By Key

map.entrySet().stream().sorted(Map.Entry.comparingByKey()).forEach(e ->
System.out.println(e));

        //Sort By Value

map.entrySet().stream().sorted(Map.Entry.comparingByValue()).forEach(e ->
System.out.println(e));
    }

}
```

```
- Java8 Streams Interview Question-26-How to sort a Map based on keys or
values using stream?-by Naren
https://www.youtube.com/watch?v=Mf7hSRZsHFs
```

Find the second largest number in a list of integers using stream ?

```
public class SecondHighestNumber {

    public static void main(String[] args) {

        List<Integer> list =
Arrays.asList(1,1,1,2,4,3,4,6,7,8,0,9,3,21,1);

        //Find the second largest number of a list

        Optional<Integer> findFirst =
list.stream().distinct().sorted(Comparator.reverseOrder()).skip(1).findFir
st();

        Integer integer = findFirst.get();

        System.out.println(integer);

    }
}
```

```
-Method overloading in depth
https://www.youtube.com/watch?v=e1QJKOGLwj4
```

```
Find Even Number Using java 8.
Find Sum of Squares Using Java 8.
Sort a List Of String Using Java 8.
Double Each Number Of The List Using Java 8.
Find distinct eleement from the list Using Java 8.
Group by Strings of equal length from the List using Java 8.
Remove String elements from List where String contains "an" using Java 8.
Join each element from the list of string to form one comma separated
String

https://www.youtube.com/watch?v=BESfZAZnABs
https://www.youtube.com/watch?v=-
ORgK_JkOa4&list=PLFGoYjJG_fqp52WKlmgF4cV72KS9_9tih
```

```
Hibernate Mapping

OneToOne
    1.User and Cart
    2.Customer and Shipping address
    - Each Customer has one shipping address.
    - Hibernate uses a shared primary key to link them.

OneToMany
    1.Order and OrderItems
    2.Product and Reviews
    - a product can have multiple reviews.
    - Hibernate uses a foreign key in the review table.

ManyToMany
    1.Product and Category
    2.Customer and Product
    -Each Customer can purchase multiple Products.
    -Products can be purchased by multiple customers.
```

```
What are the different bean scopes in spring?
1.Singleton
2.Prototype
3.Request
4.Session
5.Application
6.WebSocket
```

```
How to find second highest salary in SQL
1.max
select max(salary) from employee where salary < (select max(salary) from
employee)

2. limit

3. top

https://www.youtube.com/watch?v=a3ngELoA9h0
```

```
@RequestMapping
Difference between @PathVariable and @RequestParam In Spring Boot
Difference between @RestController and @Controller In Spring Boot
```

@Autowired Annotaion

what will happen if we write final keyword with abstract class?

- If you declare a class abstract, to use it, you must extend it and if you declare a class final you can not extend it,
- since both contradict with each other you can not declare a class both abstract and final if you do so a compile time error will be generated.

realtime example of final class provided by java?

- A real-time example of a final class in Java is the java.lang.String class,
- which is a final class and cannot be extended by any other class.
- This is because the String class is complete in nature and cannot be modified

Why Strings are Immutable or final in Java ?

1. Immutable String or Object that can not be modified once it is created. But We can only change the reference to the object.
2. The String is immutable in java because of the security,synchronization and concurrency ,caching and class loading.

Why java provided with String Constant Pool as we can store the objects in heap memory ?

- String Constant Pool provides the facility of reusability of the existing string object.
- When a new string object is created using the string literals ,then JVM first checks in the pool if this string already exists or not. if it exists ,then it will reference the existing string rather than creating a new object.
- this will help in the speeding up of the application and also helps in saving the memory as no two objects will have the same content.

What is public static void main(String[] args) ?

```
public - this method is accessible from outside the class.
static - No object is required to call the main method.
void - this method does not return any value.
main - Special method name to start the java program execution.
String[] args - the arguments passed while running the program.
```

Can we change the order of public , static , and void keywords ?

- Yes can change the order of public and static keywords , but void should come just before the method name.

how memory management or Garbage collection works in java?

- Garbage collection
  - Java provides automatic memory management through a program called Garbage collector.
  - "Remove objects that are not used anymore."
  - live object = reacheable (referenced by someone else)
  - dead object = unreachable (not referenced from anywhere)
  - Garbage collection is carried out by a daemon thread called "Garbage collector"
  - we can not force gc to happen (System.gc())

Why abstract class has constructor even though we cannot create object?

- We cannot create an object of abstract class but we can create an object of subclass of abstract class.
- When we create an object of subclass of an abstract class, it calls the constructor of subclass.
- This subclass constructor has super in the first line that calls constructor of an abstract class.
- Thus, the constructors of an abstract class are used from constructor of its subclass.
- If the abstract class doesn't have a constructor, a class that extends that abstract class will not get compiled.

What is the command for running the jar file?

```
    java –jar yourfile.jar
```

What is the command to build the project using Maven?

```
    mvn package
    mvn clean package

    This command will compile the source code, run tests, and package the
project into the specified format (usually a JAR file).
```

Wap to print 1 to 100 numbers using Stream Api

```
public class PrintOneToHundread {

    public static void main(String[] args) {

        // Wap to print 1 to 100 numbers using Stream Api

        IntStream.range(1, 101).forEach(x->System.out.println(x));

    }
}
```

MultiThreading

Different ways of creating a thread in Java

1. One is by **extending java.lang.Thread class**
2. Another is by **implementing java.lang.Runnable interface**

Why to override run() method while creating the thread using the Runnable interface ?

```
because Runnable is interface and it has abstract method called run.
Hence when we implement the Runnable interface we have to override run()
method.
```

Can We Start Thread Twice ?

- No. You can not start thread twice
- This will result in a IllegalThreadStateException

```
public class StartTwice{
    public static void main(String[] args){
        Thread t = new Thread();
        t.start();
        t.start();
    }

}
```

What methods thread use to communicate with each other ?

- Thread use wait(), notify(), and notifyAll() to communicate with each other.

Can we use POST instead of PUT and vice versa ?

- Yes, But recommendadation is to follow guidlines
- if you are requirement is to create resource then go for POST
- if you are requirement is to Update resource then go for PUT PATCH

```
https://www.youtube.com/watch?v=drwOMR3O0zc
```

Serialization and DeSerialization

- Serialization is a mechanism of converting the state of an object into a byte stream.
- Deserialization is the reverse process where the byte stream is used to recreate the actual Java object in memory.

> Note: The byte stream created is platform independent. So, the object serialized on one platform can be deserialized on a different platform.

Why we need Serialization in java?

- To transfer objects through a network.
- To keep Java objects in memory.
- To save Java objects in files.

How to implement serialization in java?

1. For **serializing** the **object**, we call the **writeObject()** method of **ObjectOutputStream**, and for **deserialization** we call the **readObject()** method of **ObjectInputStream** class.
2. We must implement the **Serializable interface** for serializing the object.

What is Serial Version UID?

- every serializable class/object gets associated with a unique identification number **provided by the JVM** of the host machine.This Unique ID is called Serial Version UID.
- This UID is used as an identification by the JVM of the receiving end to confirm that the same object is being DeSerialized at the receiving end.

What is Transient Keyword?

- Transient Keyword is a reserved keyword in Java.
- It is used as a variable modifier at the time of the Serialization process.
- **Declaring a variable with Transient keyword avoids the variable from being Serialized.**
- During deserialization, the transient variable will be assigned a default value, typically null for reference types or 0 for primitive types.

Program to find the minimum (or maximum) element of an array

```
public class MaxAndMinElementArray {

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

```
        int a[]={1,423,6,46,34,23,13,53,4};

        int max = a[0];

        for(int i = 1;i<a.length;i++) {
            if(a[i] > max) {
                max = a[i];
            }
        }

        System.out.println(Arrays.toString(a));

        System.out.println("Maximum Element is : " + max);

        int min = a[0];

        for(int i=1 ;i < a.length ; i++) {
            if(a[i] < min) {
                min = a[i];
            }
        }
        System.out.println("Minimum Element is : " + min);
    }
}
```

How to reverse the array elements in Java

```
public class ReverseArray {

    public static void main(String[] args) {

        int [] arr = {1,3,5,7,4,8,9};

        System.out.println("before Reverse : "+ Arrays.toString(arr));

        reverseArray(arr);

        System.out.println("after Reverse : "+ Arrays.toString(arr));
    }

    private static void reverseArray(int[] arr) {

        int i = 0;
        int j = arr.length-1;

        while(i<j) {

            int temp = arr[i];
            arr[i] = arr[j];
            arr[j] = temp;
```

```
            i++;
            j--;
        }
    }
}
```

Find Sub Array Of An Array

```
public class SubArray {

    public static void subArray(int arr[]) {

        int len = arr.length;

        for (int i = 0; i < len; i++) {
            for (int j = i; j < len; j++) {
                for (int k = i; k <= j; k++) {
                    System.out.print(arr[k] + " ");
                }
                System.out.println();
            }
        }

    }

    public static void main(String[] args) {

        int arr[] = { 1, 2, 3, 4, 5 };

        subArray(arr);
    }
}
```

```
Output :

1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
2
2 3
2 3 4
2 3 4 5
3
3 4
3 4 5
4
```

```
4 5
5
```

Merge Sorted Array

```
public class MergeSortedArray {

    public static void main(String[] args) {

        int[] arr1 = {1,2,3,5,7};
        int[] arr2 = {3,5,7,9};

        int [] result = mergeSorted(arr1,arr2);

        System.out.println(Arrays.toString(result));

    }

    private static int[] mergeSorted(int[] arr1, int[] arr2) {

        int len1 = arr1.length;
        int len2 = arr2.length;

        int [] result = new int[len1+len2];

        int i=0,j=0,k=0;

        while(i< len1 && j< len2) {

            if(arr1[i]<arr2[j]) {
                result[k] = arr1[i];
                i++;
                k++;
            }else {
                result[k] = arr2[j];
                j++;
                k++;
            }
        }

        while(i<len1) {
            result[k++]=arr1[i++];
        }

        while(j<len2) {
            result[k++]=arr2[j++];
        }


        return result;
```

```
    }

}
```

Find Missing Number in Array

```
public class MissingNumberInArray {

    public static void main(String[] args) {

        // Array should not have duplicates
        // Array No need be Sorted order

        int [] arr = {1,2,3,5};

        // 1+2+3+5=11

        int sum1 = 0;
        for(int i = 0 ; i< arr.length; i++) {
            sum1 = sum1 + arr[i];
        }
        System.out.println("Sum of Elements in Array ="+sum1);

        // 1+2+3+4+5 = 15
        int sum2=0;
        for(int i =1;i<=5;i++) {
            sum2 = sum2 +i;
        }
        System.out.println("Sum Of Range Of Elements in Array ="+sum2);

        System.out.println("Missing Number is = "+(sum2-sum1));
    }
}
```

Write java program to print duplicate elements of array

- input = {4,2,3,2,1,3,4,5};

```
public class DuplicateElementArray {

    public static void duplicateElement(int [] arr) {

        int len = arr.length;

        for(int i=0; i<len; i++) {

            for(int j = i+1; j<len; j++) {

                if(arr[i]==arr[j]) {
```

```
                    System.out.println(arr[i]);

                }
            }
        }
    }

    public static void main(String[] args) {

        int [] input  = {4,2,3,2,3,1,3,4,5};

        duplicateElement(input);
    }
}
```

count number of duplicate characters in a string java using Map

- https://www.youtube.com/watch?v=SGn30pD1Ryg

```
public class DuplicateCharCount{

    public static void main(String[] args){

        String str = "hello";

        duplicateChar(str);

    }

    public static void duplicateChar(String str){

        HashMap<Character,Integer> map = new HashMap<>();

        // Convert String to Char Array
        char[] ch = str.toCharArray();

        for(char c:ch){
            if(map.containsKey(c)){
                map.put(c,map.get(c)+1);
            }else{
                map.put(c,1);
            }
        }
    }
}
```

Java Program To Find Longest Substring Without Repeated Character

```
- Input  : abbac
- Output : bac --> Length is 3

- Input  : abcabcbb
- Output : abc --> Length is 3
```

```
public class LongestSubStringLength {

    public static void main(String[] args) {

        String input = "abbac";

        String longestSubstring=null;
        int longestSubstringLength = 0;


        Map<Character, Integer> map = new LinkedHashMap<Character,
Integer>();

        char[] arr = input.toCharArray();

        for(int i = 0; i< arr.length ; i++) {
            char ch = arr[i];
            if(!map.containsKey(ch)) {
                map.put(ch, i);
            }else {
                i = map.get(ch);
                map.clear();
            }
        }

        if(map.size() > longestSubstringLength) {
            longestSubstringLength = map.size();
            longestSubstring = map.keySet().toString();
        }

        System.out.println(longestSubstring);

    }

}
```

Write a function to find the longest common prefix string amongst an array of strings.If there is no common prefix ,return an empty string "".

```
Input : strs = ["flower","flow","flight"]
Output : "fl"
```

```
Input : strs = ["dog","racecar","car"]
Output : ""

https://www.youtube.com/watch?v=K5I7aUK9LVU
```

```java
public class LongestCommonPrefix {

    public static void main(String[] args) {


        String [] strs = {"flower","flow","flight"};

        String result = strs[0];

        for(int i = 1; i< strs.length ; i++) {
            result = common(result, strs[i]);
        }

        System.out.println(result);

    }


    public static String common(String s1,String s2) {

        int n = Math.min(s1.length(), s2.length());

        StringBuilder sb = new StringBuilder();

        for(int i = 0 ; i<n ;i++) {

            if(s1.charAt(i)== s2.charAt(i)) {
                sb.append(s1.charAt(i));
            }else {
                break;
            }
        }
        return sb.toString();
    }
}
```

How to make read only ArrayList in java.

- Use Collections.unmodifiableList() method.

```java
public class ReadOnlyArrayList {

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

```
        List<String> list = new ArrayList<>();

        list.add("Apple");

        list.add("Banana");

        list.add("Mango");

        System.out.println(list);

        List<String> readOnlyList = Collections.unmodifiableList(list);

        readOnlyList.add("Kiwi"); //UnsupportedOperationException
    }
}
```

Write a java program to find the average salary of each department using java 8 Stream API?

```
public class AverageSalaryByDepartment {

    public static void main(String[] args) {

        List<Employee> employees = Arrays.asList(
                new Employee("John", "HR", 50000),
                new Employee("Jane", "IT", 60000),
                new Employee("Doe", "IT", 70000),
                new Employee("Smith", "HR", 55000),
                new Employee("Alice", "Finance", 75000),
                new Employee("Bob", "Finance", 80000)
        );

        Map<String, Double> averageSalaryByDepartment = employees.stream()
                .collect(Collectors.groupingBy(Employee::getDepartment,
                        Collectors.averagingDouble(Employee::getSalary)));

        System.out.println("Average Salary by Department:");

        averageSalaryByDepartment.forEach((department, averageSalary) ->
                System.out.println(department + ": " + averageSalary));
    }
}
```

sort an employee list using the Stream API in Java

```
public class Main {
    public static void main(String[] args) {
        List<Employee> employees = new ArrayList<>();
        employees.add(new Employee("John", 30, 50000));
```

```
        employees.add(new Employee("Alice", 25, 60000));
        employees.add(new Employee("Bob", 35, 45000));

        // Sorting employees by name
        List<Employee> sortedByName = employees.stream()
                .sorted(Comparator.comparing(Employee::getName))
                .toList();

        System.out.println("Employees sorted by name:");
        sortedByName.forEach(System.out::println);

        // Sorting employees by age
        List<Employee> sortedByAge = employees.stream()
                .sorted(Comparator.comparingInt(Employee::getAge))
                .toList();

        System.out.println("\nEmployees sorted by age:");
        sortedByAge.forEach(System.out::println);

        // Sorting employees by salary
        List<Employee> sortedBySalary = employees.stream()
                .sorted(Comparator.comparingDouble(Employee::getSalary))
                .toList();

        System.out.println("\nEmployees sorted by salary:");
        sortedBySalary.forEach(System.out::println);
    }
}
```

find out all the numbers starting with 1 in a list by using Stream API ?

```
    List<Integer> list = Arrays.asList(20,15,80,11,48,25,98,32,17);

    list.stream().map(s -> s.toString()).filter(s ->
s.startsWith("1")).forEach(System.out::println);
```

Write a java program to find sum of even numbers and sum of odd numbers in a given list using java 8 streams.

```
List<Integer> list = Arrays.asList(3,5,6,8,9);
```

```
public class SumOfEvenOdd {

    public static void main(String[] args) {
        List<Integer> list = Arrays.asList(3,5,6,8,9);
```

```
        //filter even numbers and add to list
        List<Integer> listOfEvenNum = list.stream().filter(n -> n % 2 ==0
).collect(Collectors.toList());
        System.out.println(listOfEvenNum);

        int sumOfEvenNumbers = list.stream().filter(n-
>n%2==0).mapToInt(Integer::intValue).sum();

        System.out.println(sumOfEvenNumbers);

        //filter Odd numbers and add to list
        List<Integer> listOfOddNum = list.stream().filter(n->n%2 ==
1).collect(Collectors.toList());

        System.out.println(listOfOddNum);

        int sumOfOddNumbers = list.stream().filter(n-
>n%2==1).mapToInt(Integer::intValue).sum();

        System.out.println(sumOfOddNumbers);
    }
}
```