Message
###for java
from 15th to 20th feb -revise whatever i completed and worked on since 30th dec ,everything - code which i created , any concept doc or file in work laptop or in google coolab , using intellig , in git , from harsith solved prolem ,WhatsApp group , personal laptop java file
from 21st to 23rd- whatever you have written and shortlisted , or concept not much clear or part which i need to memorise like iteration through linked list , queue, some more concept or way or writing or handling like exception handling , which treeset or array to use , matrix concept , concept of recusion and dynamic programming, some basic multple choice basic java programming , concept of overriding , overloading , file reading method and funciton , abstract class and interface
24th- just go through all the marked and last minute remembered concept
join(delimiter, elements...)-Use when you need to join words with a delimiter
"System.out.println(str.matches(""[a-z]+\\d+""));"
Arrays.toString(array) – Print an Array
Arrays.binarySearch(array, key)
Arrays.sort(array)
Arrays.copyOf(array, newLength)
Arrays.asList(arr)
List<Integer> list = new ArrayList<>(Arrays.asList(1, 2, 3)); Integer[] arr = list.toArray(new Integer[0]);
when daling with count of character ,use int[] count = new int[26];
count[c - 'a']++;
countMap.put(c, countMap.getOrDefault(c, 0) + 1)
Finding the First Non-Repeating Character-use hashmap to store count and then return once u get first char with just 1
arrays-store in set or map and sometimes use another set to put when iterating through arrays , also for finding unique element , to find pairs for sum
use concept of sum and rem to find another elem
finding somethiig between 2 arrays can be solved by using set , once to put in set and then iterate over other arrays to check if element exist in ser
for (int num : arr) { if (seen.contains(num)) return num; seen.add(num); } this concept can be used to iterate over same arrays multiple times
for dynamic programming , the default case is also stored in array like if n==0 or 1 , its 1 then a[0]=a[1]=1 and then run loop on remaining posiiton of n and use function given in question to derive elements in another position , the return should be a[i]
matrx- see which part should be in j and how to done across each row or each column based on that do operation and store in some collecton if applicable
you can also use int temp = matrix[i][j]; like concept to shift or arange element
"List Operations Sorting, Searching, Removing duplicates Set Operations
Finding unique elements, Union, Intersection, Difference Map Operations Frequency counting, Grouping, Caching"
List<Integer> numbers = Arrays.asList(5, 2, 8, 3, 2, 8); Set<Integer> uniqueNumbers = new HashSet<>(numbers);
set.add(num)- return false when there is already num in set

Interfaces cannot have constructors. They can have abstract methods and default methods.
Usage of @Override: If a subclass extends an abstract class and provides an implementation for an abstract method, you must use the @Override annotation to indicate that the method is being overridden from the parent abstract class.
If a class implements an interface and provides a custom implementation for a default method, you can (and should) use the @Override annotation to explicitly indicate that you're overriding the default method.
"interface Animal { default void sound() { System.out.println(""Some sound""); } } class Dog implements Animal { @Override public void sound() { System.out.println(""Woof!""); } }"
Character.toString(ch) - Convert a character to a string
"char[] charArray = {'h', 'e', 'l', 'l', 'o'}; String str = new String(charArray); // ""hello"""
Division (/) and modulus (%) are your main tools for manipulating digits.
while (num != 0) { int digit = num % 10; // Extract the last digit reversed = reversed * 10 + digit; // Shift reversed digits and add the new one num /= 10; // Remove the last digit from num }
for number if asked to chec if palindrome, check with reverse of the digit
You can use powers of 10 to extract specific digits. For example, to get the third digit from the right (in a 4-digit number), use division: num / 100 % 10
When solving string manipulation problems that involve both letters and digits, you need to focus on distinguishing the two types of characters and applying the right operations accordingly.
Manipulating digits: When you need to manipulate digits, you can convert them to an integer (if needed) using Character.getNumericValue() or by converting the digit character to an integer using Integer.parseInt()
If you need to separate letters and digits, consider using a loop to iterate through the string and check whether each character is a letter or digit
Understand the grouping mechanism with parentheses () and alternation with the pipe | symbol in regular expression
java.util, java.io, java.math
###stragtegy for concept and wording asked in ques
1.looping through 2 scalar vector , which is in array form , find sum doing scalar peoduct , use sum variable to find final sum , page 1 (merged ques2)
ques- finding first or last value for diff datatype like input number ,using while(true)
patterns-question ,The pattern consists of an n × n grid. Each row starts with X characters, followed by o characters. The number of Xs decreases, while the number of os increases as the rows go down.
to add new key u don't have to loop through existing keys , since in beginning there is no keys , it would never enter the loop, looping through key should not be done, if you want to add new keys and value , like for this example if(records.containsKey(name)) {}
break Statement Exits the loop immediately (whether it's a for or while loop).
"for concat String result = str1.concat("" "").concat(str2), u should save in new string after concatention"
"logger.warning(""Unauthorized access"")"

```
import java.util.logging.Logger;
public static boolean isDouble(int value) { return (value/10==value%10);
} when alsed if number is dublet like 11,22
"for this ""  Entry leastLearned = null;  // Find the least learned entry
for (Entry entry : entries) { if (entry != null && (leastLearned == null
|| entry.getLearned() < leastLearned.getLearned())) { leastLearned =
entry; } }"" ##very imp , its like find max element in array, with some
strt point Entry leastLearned = null; This initializes leastLearned to
null, meaning it does not reference any object initially. It acts as a
placeholder to store the least-learned Entry object as the
loop progresses"
can also reference to first one Entry leastLearned = entries[0]; like we
did with small ,max min value problem for array
"public String[] generalIngredients() { return new String[]{""Yeast
Dough"", ""Tomatoes"", ""Mozzarella"", ""Oregano""}; }"
"ingredientsBuilder.append(ingredient).append("", "");"
System.exit(0)
Instances of the type Professor additionally have an array assistants, in
which assistants are stored. A professor can have a maximum of 10
employees. means class professor has variable private
ArrayList<Assistant> assistants;
Instances of type Assistant also have a component supervisor
of type Professor. -private Professor supervisor;
"The statement ""Instances of type Assistant also have a component
supervisor of type Professor"" means that every object of class Assistant
contains a reference (or attribute) to an object of class Professor"
supervisor.addAssistant(this)
where a assistant quits its job, i.e. is removed from the assistants
array of its boss. public void resign() { if (supervisor != null) {
supervisor.removeAssistant(this); supervisor = null; } }
"@Override public String toString() { StringBuilder sb = new
StringBuilder(super.toString() + "", Assistants: ""); for (Assistant
assistant : assistants) { sb.append(""\n  - "").append(assistant); }
return sb.toString(); }"
public int compareTo(Pet pet) throws NotComparableException { if (!(pet
instanceof Cat)) { throw new NotComparableException(this); } return
Double.compare(this.weight, pet.weight); } ##passing and comparing this
instance
return Double.compare(this.weight, pet.weight);
"first split by , name: ""Laptop"", artno: ""A1234"", price: 1200  then
if (part.startsWith(""name:"")) { name =
part.split("":"")[1].trim().replace(""\"""", """"); }--use of srartwith
to recognise the right part"
"String[] parts = input.split("","",""); for (String part : parts) { if
(part.startsWith(""name:"")) { name =
part.split("":"")[1].trim().replace(""\"""", """"); } }"
try (BufferedWriter bw = new BufferedWriter(new FileWriter(modFileName)))
{ for (Product product : products) { bw.write(product.toString());
bw.newLine(); } } catch (IOException e) { e.printStackTrace(); }
products.add(new Product(line));-- add product type
"int seed=12345;  Random r=new Random(seed)   for (int num : arr) { if
(num > max) { secondMax = max; max = num; } else if (num > secondMax &&
num != max) { secondMax = num; } } return new int[]{max, secondMax}; }
int min = Math.min(a, Math.min(b, c)); int max = Math.max(a, Math.max(b,
```

c)); System.out.format(""| %6.2f | %7.2f | %7.2f |\n"", side, area, volume); for above result 6: Specifies that the total width of the number (including the digits, decimal point, and spaces) should be at least 6 characters. If the number is shorter than 6 characters, it will be padded with spaces on the left String formattedTime = String.format(""%02d:%02d:%02d"", hour, minute, second); String ampm = (hour >= 12) ? ""PM"" : ""AM""; System.out.format(""| %4d | %6d | %6d | %-8s |\n"", hour, minute, second, formattedTime + "" "" + ampm); result String.format to use when want to represent multiple variable together like in time , everything in one line like 02:03:34 Also in this formatted string can be used inside System.out.format as one single string %02d: The 0 means that if the number is smaller than 2 digits, Java will pad it with a leading zero to make it exactly 2 digits wide. there is also some question, where its while loop with index and the index value is either updated by array value or by some logic like in question where to find cyclic or just traverse through array based on array values. like this current = array[current]; for (char c : s.toCharArray()) { charCount[c]++; } --directly incrementing counter based on index value, this concept is used while cal no of character Character ch = 'A'; // Using Character object String d = ch.toString(); if (n == 1) return 2.0, sometimes returning double char use a cache (e.g., array, map) to store already computed results to avoid recomputation. in dynamic programming for dynamic, default case with return alongwith array fill F(n)=F(n-1)+F(n-2) here f is function and n-1 is arguments so in recursvice and dynamic recusive , F becomes function in dynamic , F goes to become Array like a[i] where i is argument and it has to be filled in for loop sometimes for dynamic inside loop , finding value of new index can be with i like for(int i=2;i<=n;i++) { b[i]=((i))*b[i-1]; } n case of dynamic programming b[i]=(b[i-1]+b[i-2]); and in case of recusion return (frec(n-1)+frec(n-2)); System.out.println((int) 'A'); -gives 65 System.out.println((char) 65); -gives A b[i]=(char)((int)c[i]+shift); - shifting char by adding some digit this.content = new StringBuilder(str); new ArrayList<>(): Creates an empty list. new ArrayList<>(Arrays.asList(...)): Initializes with elements. // Empty set Set<String> set2 = new HashSet<>(Arrays.asList(""A"", ""B"", ""C"")); // Pre-filled set this refers to the current object instance aString.regionMatches(true, 13, ""KRINGEL"", 2, 3); Explanation of the Arguments: aString: This is the first string where you are specifying a region to compare. true: This means the comparison is case-insensitive. 13: This is the starting index in aString for the region to compare. ""KRINGEL"": This is the second string against which the region of aString is being compared. 2: This is the starting index in ""KRINGEL"" for its region to compare. 3: This is the number of characters to compare from both regions. String str = new String ( ); String str1 = new String (""a String""); String str2 = new String(new char[] {'a', 'b', 'c'}); char[] charArray = {'H', 'e', 'l', 'l', 'o'}; String str = new String(charArray); char[] charArray = {'H', 'e', 'l', 'l', 'o', ' ', 'W', 'o', 'r', 'l', 'd'}; String str = new String(charArray, 0, 5); // ""Hello"" int[] arr; arr = new int[]{1, 2, 3, 4, 5}; --when doing in 2 line , new int[] shiuld be there This syntax is valid because it explicitly creates a new array object of type int[] and assigns it to the variable arr. The new keyword provides the necessary context for the compiler to understand that an array is being created When you're assigning an array to a variable that has

already been declared, you must use the new keyword to explicitly create the array object. Math.random() Generates a random double value between 0.0 (inclusive) and 1.0 (exclusive). 2. Math.random() * 6 Multiplies the random value by 6, giving a result in the range [0.0, 6.0). 3. (int) (Math.random() * 6) Casts the result of Math.random() * 6 to an integer, truncating the decimal part. This gives an integer in the range [0, 5] frequency [1 + (int) (Math.random() * 6)] --again using some variable in index for (int roll = 1; roll <= 300; roll++) frequency [1 + (int) (Math.random() * 6)] ++; way to have count of array with fixed size , by in for loop , since index out wud have value from 0 to 6 only if (Character.isLetterOrDigit(c)) --can be used for removing punctuation enum Weekdays { MONDAY, TUESDAY, WEDNESDAY, 3 THURSDAY, FRIDAY, SATURDAY, SUNDAY }; Weekdays aDay = Weekdays.TUESDAY; for (String line : lines) { 10 nbw.write(line, 0, line.length()); 11 nbw.newLine(); Files.write(path, updatedLines);, List<String> lines = Files.readAllLines(path);, writer.write(data); Path path = Paths.get(""example.txt""), Files.write(path, updatedLines); import java.io. File file = new File(filePath) try-with-resources - It is a specialized form of the try block that is used for managing resources like files, streams, or database connections that implement the AutoCloseable interface. To Append to the File Instead of Overwriting: You can use the overloaded constructor of FileWriter that takes a second boolean parameter: FileWriter f = new FileWriter(filename, true); --for appending file Write back integers to file: Ensure that when you write the even numbers back to the file, you properly write them as strings, as Files.write() expects a list of strings --imp Read file as integers: Since readAllLines() returns a list of String objects, we need to convert them to Integer before processing them To correctly write an integer to a file, you need to convert the integer to a string first, or use BufferedWriter, which is better suited for writing primitive types if (!Files.exists(p)) { // Create the file Files.createFile(p); File f=new File(filename); if(!(f.exists())) { f.createNewFile(); } public static void writeFile(String filePath, List<String> data) throws IOException { try (BufferedWriter writer = new BufferedWriter(new FileWriter(filePath))) { for (String line : data) { writer.write(line); writer.newLine(); // Add a new line after each entry } } } nbw.write(line, 0, line.length()); nbw.write(line, 0, line.length()): Writes a portion of the string line to the file. line is the string to write. 0 is the starting index (beginning of the line). line.length() is the length of the string (entire line is written) ""\\b[A-Z][a-z]*\\b""; // Words starting with a capital letter String regex = ""\\b\\w*at\\b""; // Words ending with 'at' it ensures that 12-12-2023 is matched, but abcd12-12-2023 is not. String regex = ""\\b(0[1-9]|[12][0-9]|3[01])-(0[1-9]|1[0-2])-(\\d{4})\\b""; (0[1-9]|[12][0-9]|3[01]) This part matches the day (DD): 0[1-9]: Matches days 01 to 09. [12][0-9]: Matches days 10 to 29. 3[01]: Matches days 30 or 31. --imp Without \\b Regex: (0[1-9]|[12][0-9]|3[01])-(0[1-9]|1[0-2])-(\\d{4}) ""abc12-12-2023xyz"" (matches the 12-12-2023 part of the string, even though it's embedded). System.out.println(str1.compareTo(str2)); indexOf(int ch, int fromIndex) - Find the index of 'J' after index 10 int indexOfJFromIndex = str.indexOf('J', 10);   // 6. lastIndexOf(int ch, int fromIndex) - Find the last occurrence of 'J' before index 30 id and pw are associated. Pay attention to what initial values the individual array components id[i]

and pw[i] are set! public UserManagement() { Arrays.fill(id, null); Arrays.fill(pw, null); }"
reverse of array can also be done by using concept of swapping where , start is swappend with end and both counter are adjusted to swap next number
Use indices or array slices to handle subarrays in recursive calls. For example, you can pass the start and end indices for the current subarray to the recursive function
List<List<Integer>> subsets = findSubsets(arr, 0, sum, new ArrayList<>()); sending empty arraylist
emember that in a 2D matrix matrix[i][j], i refers to the row index and j refers to the column index.
p[j][i]=m[row-(i+1)][j]; --bring last row to zero you are shifting matrix by 90 clockwise for anticlockwise ,p[i][j]=m[j][cols-(i+1)];
atleast try with 2 example for coming with logic, se do u have to traverse through rows or cols
The line k = k % n; // Normalize rotation count ensures that the number of rotations k is adjusted to a valid range within the length of the string (n).
sometimes, when they ask for some output , like beep , with array of both , just print it . don't store in another array
while (number != 0) { int digit = number % 10; // Extract the last digit reversed = reversed * 10 + digit; // Add it to the reversed number number /= 10; // Remove the last digit }## reverse of digit
field[i] = random.nextInt(21) - 10; // Random numbers from -10 to 10 when put(20) means from 0 to 19
Write comments explaining your logic if time runs out
Infinite Loops in Recursion/Loops: Solution: Ensure base cases are correct and loops have termination conditions.
public void enqueue(int data) { Node newNode = new Node(data); if (rear == null) { // If queue is empty front = rear = newNode; return; } rear.next = newNode; rear = newNode; }
The loop stops when temp.getNext() is null, meaning the last element is not processed in this loop. This works only if you don’t need to process the last element
o find the smallest positive int number n that is so large that the value n · n can no longer be represented as a int means to find this value , if value goes beyond this it would be long
in linked list   Node newNode = new Node(name, age); is based on constructor of Node/elem, node can have more than 1 info like age, name
m[i] where if i=1 it gives first array of matrix , in this way cols of matrix can be initlized dynamically like below int[][] result = new int[a.length][]; for (int i = 0; i < a.length; i++) { if (a[i] < 0) { // Ignore negative sizes (could also throw an error) result[i] = new int[0]; } else { result[i] = new int[a[i]]; // Initialize inner arrays with a[i] size } }
int[] a = {3, 5, -2, 4}; // Defines column sizes for each row int[][] result = new int[a.length][];
Understanding m[i] in a 2D Matrix (Jagged Array)
// Loop through each word for (int i = 0; i < words.length; i++) { if (!seen[i]) { // Check if this word is already marked uniqueCount++; // Count it as unique // Mark all occurrences of this word for (int j = i + 1; j < words.length; j++) { if (words[i].equals(words[j])) { seen[j] = true; // Mark duplicates } } }

// Constructor to initialize the list of products public Evaluation(List<Product> products) { this.products = products; }
##initialize list with another list
Make sure that, as usual, the rating is in the interval between 0 and 5 inclusive. If a value is less than 0, set the rating to 0 and if the rating is greater than 5, set the rating to 5 it says handle this situation when initilizing the rating column , if you separately call it , then it won't handle all the time
both are similar //Restaurant b=new Restaurant(restaurant[0].name,restaurant[0].rating); Restaurant b=restaurant[0];
"parts[0].matches(""\\d{5,}"")) // ID must have at least 5 digits"
when asked these variable is of object type so also declared as object
Object key; Object value; public DictionaryElement(Object key, Object value)
for (int i = 0; i < dictionary.length; i++) { if (dictionary[i] == null) { dictionary[i] = new DictionaryElement(key, value); flag = true; } else { if(dictionary[i].getkey() == key) { dictionary[i].setvalue(value); //here can directly return true; flag = true; } }---when need to overwrite key
sometimes directly return true or false is better than using flag=true , which give smetimes diff result , so if get any issue directly return true or false
"in this public static void main(String[] args) { // Test the PhoneBook functionality PhoneBook.change(""Alice"", ""123456""); } we directly call method using classname since all members of phonebook is static as mentioned in question Your taks is to create a PhoneBook class that contains only static class members. this means all are called using calssname"
Yes, if all members of a class are static, there is no need for a constructor because constructors are used to initialize instance (non-static) members of a class. Since static members are tied to the class itself and not to any specific instance, you can work with them without creating objects of the class
###very imp to have break when asked for infinte loop Using while(true) in Java is appropriate when you need an infinite loop and don't know beforehand how many iterations are required. It allows you to repeatedly execute a block of code until a specific condition is met, at which point you can break out of the loop. However, it should always include a clear exit condition (e.g., using break or return), or it will lead to an infinite loop
Integer obj = 42; // Convert Integer object to int int value = obj.intValue();
"In Java, once an array is created, its size cannot be changed. This means that you can't directly resize an existing array. However, you can create a new array with a different size and assign it to the variable (like this.data = new int[array.length];), effectively ""replacing"" the old array with a new one of the desired size. whichhas been done in above code"
either you have return or break but not both together if (p.name.equals(productName)) { return p; // break; }
in void method also , under some if condition , we can call return keyword to end method .its better

"find no of cat in string public static int countCat(String str) { // Base case: If length of string is less than 3, ""cat"" can't exist if (str.length() < 3) { return 0; } // Check if the first three characters are ""cat"" if (str.startsWith(""cat"")) { return 1 + countCat(str.substring(1)); // Move one step ahead } else { return countCat(str.substring(1)); // Move one step ahead } } ##recusrion where , the base class is based on str.length"
"System.out.printf(""%-10s: %.2f°C%n"", months[i], avgTemperatures[i]);-- any char or symbol in inside """""
Answer: == checks reference equality, while .equals() checks logical equalit
"String a = ""rajan""; String b = new String(""rajan"");
System.out.println(a == b); // false (different memory locations)
System.out.println(a.equals(b));"
value is list of string in this map, calling get and then using add to load into list public Map<String, List<Movie>> getGenreMap() { Map<String, List<Movie>> genreMap = new HashMap<>(); for (Movie movie : m) { for (String genre : movie.getGenres()) { // If genre is not present, add an empty list genreMap.putIfAbsent(genre, new ArrayList<>()); // Add movie to the list for this genre genreMap.get(genre).add(movie); } } return genreMap; }
replace(seq,newseq) - this function is also imp where some char need to be replaced
when in question said , defined variable like left and right, should be able to store any data type available in java --very tricky question -- solution very imp public class Tuple<T, U> { private static int idCounter = 0; // Auto-incrementing ID private int id; private T left; private U right;
In Java, Object is the superclass of all classes, meaning left can store any type of value (e.g., Integer, String, Double, List<Integer>, etc.). However, it lacks type safety, requiring explicit type casting when retrieving values.
"Tuple<String, Integer> t1 = new Tuple<>(""A"", 10); because class is Tuple<T, U>"
in case when each row has diff length of column avg[i] = sum/temperatures[i].length; temp[i] is one row
// Check which elements of the second array are in the HashSet for (int num : arr2) { if (set.contains(num)) { common.add(num); set.remove(num); // To avoid duplicates in the result } }
public static int[][] createArray(int[] a){ int[][] res = new int [a.length][]; for(int i=0;i<a.length;i++){ res[i] = new int[a[i]]; } return res; }
if(tripsremaining>0){ tripsremaining--; }## good way to reduce counter when it haooens
"public Post01(String content, String hashtags) { this.content = content; String[] str = hashtags.split(""\\s+""); this.hashtags = new ArrayList<>(Arrays.asList(str)); }##splitting string into string array and storing in list inside constructor"
in the question,it was asked The constructor should store the hashtags as a list of individual hashtag string
"for (Post01 post : posts) { for (String tag : post.getHashtags()) { hashTagCount.put(tag, hashTagCount.getOrDefault(tag, 0) + 1); } } hashTagCount.forEach((tag, count) -> System.out.println(tag + "": "" + count));"

in construcot, the object which is passed , can be asked to manipulate it and initilaize some local vairable too
in startwith ,i can also put index startsWith(search, i) so can check some pattern in string
int index2=c.charAt(i); gives This character is then implicitly converted to an int, which means it returns the ASCII
String.valueOf(a.charAt(i)) converts the char to a String first, which is unnecessary when appending to StringBuilder, since StringBuilder.append(char) already handles char
instead of (char) (i) + 'a', use (char) ('a' + i),##imp
public Human(String name) { this.name = name; this.id = nextId++; this.father = null; this.mother = null; } initlalize with null --to have some value for all of the objects of class even when its not passed
public SimplePlaylist(int capacity) { songs = new Song[capacity]; count =0; } public void addSong(Song song) { if (count < songs.length) { songs[count++] = song; } public String play(String title) { for (int i = 0; i < count; i++) { if (songs[i].getTitle().equals(title)) { return songs[i].toString(); } } ##array is initialized with some capacity and then new element is added using count++ counter , which track index of array
use of substring(0,mid) substring(mid) would be one way to find char in strin g
"in string eturn , return null or """" in base case and for int recturn 0"
if <0 or > than str.length or remaining string lenght is less than target string length so ,now no comarison is possible
"Map<Integer, Student> studentMap = new HashMap<>(); studentMap.put(101, new Student(""Alice"", 101));## where value is object of some class"
private List<List<String>> employeeGroups; // Nested list (list of lists)
public Department(String name) { this.name = name; this.employeeGroups = new ArrayList<>(); } employeeGroups.add(new ArrayList<>());
employeeGroups.get(groupIndex).add(employeeName)### list of list , adding new element to index of index
for (List<String> group : employeeGroups) { if (group.contains(employeeName)) { return true; } ### iterating throug list of list and finding elements in list
"map.put(""fruits"", new ArrayList<>(Arrays.asList(""apple"", ""banana"", ""orange"")))"
"university.get(""Computer Science"").put(""Data Structures"", new ArrayList<>(Arrays.asList(""Alice"", ""Bob"", ""Charlie"")))"
"categoryMap.get(""Fruits"")-- thin which type of data or collection is returned by calling get and perform method"
cross check exception like , when choosing form recommended one , IllegalArgumentException ,match with what said in question
you have setter and getter for variables which is defined in class , not some variable which is defined in super class
"When you apply trim() directly after split("" ""), you likely encounter an issue because split("" "") returns an array of strings"
"public void post2(String content, String hashTags) { // Convert hashtags string into a list List<String> hashTagList = List.of(hashTags.split("" "")); // Create Post object Post newPost = new Post(content, hashTagList); // Add post to the list p.add(newPost); } convert string to list since one variable is of list type"

"List.of(...): Converts the array into an unmodifiable list. The resulting list cannot be changed (no add, remove, or modify operations). For example, it becomes: [""apple"", ""banana"", ""cherry""] as a lis"
"word = word.replaceAll(""[^a-zA-Z0-9#]"", """"); // Remove punctuation##very imp , in single line remving part of string"
public void enqueue(String value) { if (head == null) { head = new Elem(value); return; } Elem temp = head; while (temp.next != null) temp = temp.next; temp.next = new Elem(value); } ###in this the temp will move to last node because after this temp.next will be null , so new elem is added to last node
public static int countSubstring(String s, String t) { int count = 0; int index = s.indexOf(t); while (index != -1) { count++; index = s.indexOf(t, index + 1); } return count; }
atch (AgeTooLowException | IllegalAddressException e) -- both exception in one line
"public IllegalAddressException(String message, String address) { super(message+"" ""+address); } ##passing some variable in constructor"
Person(String name, int age, String address) throws NullPointerException,AgeTooLowException{}
"As message they pass the string ""Names are not allowed to be null"" to the exception constructor means throw new AgeTooLowException(""Age must be greater or equal to zero:"",age);"
"private static final List<String> GENERAL_INGREDIENTS = Arrays.asList(""Yeast Dough"", ""Tomatoes"", ""Mozzarella"", ""Oregano"");"
this.specialIngredients = new ArrayList<>(specialIngredients); allIngredients.addAll(GENERAL_INGREDIENTS);
if (this.getClass() != pet.getClass()) { throw new NotComparableException(pet); }
if (!(this instanceof Cat)) { throw new NotComparableException(this); } return Double.compare(this.weight, pet.weight);
public void reverse() { Node prev = null, curr = head, next; while (curr != null) { next = curr.next; curr.next = prev; prev = curr; curr = next; } head = prev; } ##imp
before updating curr node, point refe or , see which is going to become curr pointer now
Mediathek result = new Mediathek(this.name);
"Matches: Any string made up of letters, digits, or underscores in any order -s(""\\w+"") Requires: A sequence of word characters (\\w+) followed by one or more digits (\\d+)-s(""\\w+\\d+"")"
the way to return null list Collections.emptyList(); ####imp
while ((line = reader.readLine()) != null) { AddressOptimizer optimizer = new AddressOptimizer(line); List<String> result = optimizer.optimize(); }--creating object from each line of file
map.size() will give no of distinct keys---##imp for question
m.put(p.nationality,m.getOrDefault(p.nationality,0)+1); pass same arg , i was making mistake by putting p in getordefault
"input.split("",|;|\\+""); // Splits by comma, semicolon, or plus"
public SpaceAsset(double x, double y, double z) { this.coordinates = new double[]{x, y, z}; this.speed = 0.0; // Default speed }
// Find the second occurrence of s2 in s1 after the first occurrence int secondOccurrence = s1.indexOf(s2, firstOccurrence + s2.length()); ####imp
"[apple, banana, cherry] int index = example.indexOf(""banana"");
example.add(index + 1, ""star""); gives [apple, banana, star, cherry]"

public IntElem getPrev(int value) { IntElem current = start; IntElem previous = null; while (current != null && current.getValue() < value) { previous = current; current = current.getNext(); } return previous; }
replace(oldChar, newChar) - Replaces all occurrences of a character.
replaceAll(oldString, newString) - Replaces all occurrences of a substring using regex
return m.matches(); -- return in boolean
"tr.split(""[, .]+""); // Split by comma, space, or dot"
if(queue[(front+i)%capacity]!=queue[(rear-i)%capacity]) -comparing front to rear point
"for(int j=i+1;j<a.length;j++) { if(a[j]==rem) { System.out.println(""rem value matching at j ""+j +"" "" + rem); l.add(a[i]); l.add(a[j]); l2.add(l); //l.removeAll(new ArrayList<>(l)); l=new ArrayList<>(); }"
"sometimes comparing one string to concat of other 2 can solve result like rajansah equals ""Rajan""+""sah"" or sah+rajan"
for (int num : nums) { sum += num; if (map.containsKey(sum - k)) { count += map.get(sum - k); } map.put(sum, map.getOrDefault(sum, 0) + 1); }## way to manipulate keys of map and then sum values
This technique is often used when you need to work with digits represented as characters, like when you're parsing numbers from a string or processing numeric input int digit = ch - '0'; // digit will be 7, an integer
ch-'0' here ch is character ch-5-- here ch considered int
When you subtract the character '0' from num1.charAt(i), you're essentially converting the character representing a digit into its corresponding integer value.
for (int i = 0; i < rows; i++) { for (int j = 0; j < cols; j++) { for (int k = 0; k < common; k++) { result[i][j] += matrix1[i][k] * matrix2[k][j]; }
public static int pipSum(Dice d1, Dice d2) { return d1.pips() + d2.pips(); }## in this Dice d1 , this method is defined inside Dice class ,
public Mia() { d1 = new Dice(); d2 = new Dice(); }##initilizing object of diff class in this class constructor
use either of break or return to come out loop specailly in question of array
result[index++] = deQueue(); --one way to fill data in array
"String motherName = (mother != null) ? mother.getName() : ""Unknown""; --when we have to choose from 2 or do min,max , use these logic to get variables final value"
protected List<String> ingredients; , accessed in diff class and more elem is added in it
Protected Meaning: When a variable or method is declared as protected, it can be accessed: Within the same class. By subclasses (through inheritance) even if they are in different packages. By other classes in the same package
protected List<String> ingredientsList; This means that ingredientsList is unique to every instance of the Recipe (and its subclasses like Pizza, Sandwich, etc.).
When you instantiate an object of any subclass (say Pizza), a new ingredientsList is created for that specific object. It doesn't share the ingredientsList with other instances or classes, because each object gets its own copy of the list

"token.equals(""+"") || token.equals(""-"") . Stack<Double> stack = new Stack<>(); throw new Exception(""Invalid expression."");"
"Before applying an operator, test if there are enough arguments on the stack to carry out the calculation. If this is not the case, throw an Exception if (stack.size() < 2) { throw new Exception(""Not enough operands.""); }"
after clac result, pusing into stack, stack.push(result);
public int compareTo(Pet pet) { if (this.weight < pet.weight) return -1; if (this.weight > pet.weight) return 1; return 0; } ## this object is present and can be accessed
public abstract int compareTo(Pet pet) throws NotComparableException; ##abstract class throws Exceptio
the instanceof operator is used to check whether an object is an instance of a specific class or a subclass thereof
Cat cat = (Cat) pet; // Safe cast
sometimes in custom exception, more method can be declared like public String getErrMsg() {} , also some variable can be calc
in custom , some variable which is calc there , can be passed in custom return method
"public Person(String name, int age, String address) throws NullPointerException, AgeTooLowException, IllegalAddressException { if (name == null) { throw new NullPointerException(""Names are not allowed to be null""); } if (age < 0) { throw new AgeTooLowException(""Age must be greater or equal to zero:"", age); } if (address == null || !address.contains("","")) { throw new IllegalAddressException(""Address is not correctly formatted:"", address); }"
"public AgeTooLowException(String message, int age) { super(message + "" "" + age); --constructor passes in string }"
final class AddressUtil {- to access method of this class , create object of this class and then call it, if its in diff package , then need to import this class and method in another package
Use find() when you want to search for occurrences of a pattern within a larger string, Searches the input string for any substring that matches the pattern matcher.matches()- Checks if the entire input string matches the given regular expression.
".split(""\\s"", 2): Splits the address string at the first occurrence of any whitespace charact. 2: The limit parameter, which controls the maximum number of splits. Why Use limit = 2? It ensures that the rest of the string is kept together in the second element, no matter how many additional spaces or words are present. This is useful when you only need to separate the first word or number from the rest of the string"
Arrays.toString(parts2) part2 is string[]
Limit = 3: Splits at the first two spaces. [12345, Berlin, Germany Europe] First part: 12345 Second part: Berlin Third part: The rest
\\s matches exactly one whitespace character , when use +- its 1 0r more so when no symbol used, its just that presence
Collections.addAll(result, parts); -add one cllection , parts to result
"/* Scavenger hunt in the array */ index = 0; counter = 0; while (counter < 10) { System.out.print(index + "" ""); index = array[index]; // Move to the next index counter = counter + 1; }"
while (index < size - 1) { if (array[index] > array[index + 1]) { inversions = inversions + 1; }}- some kind of counter , to count when certain condition is fulfilled
array[i] = rand.nextInt(limit + 1); // Random number between 0 and limit

public static boolean prime(int num) { if (num <= 1) return false; for (int i = 2; i <= Math.sqrt(num); i++) { if (num % i == 0) return false; } if (testX.toString().equals(testY.toString())) return 1;
// Static method to set publishing public static void setPublishing(String publishing) { Book.publishing = publishing; } --book is class
leastKnown.setLearned(leastKnown.getLearned() + 1);-updating variable by fetching it
"sb.append(entry.toString()).append(""\n""); ..one way to have string is result from one class"
in one tostring, calling tostring of other class
There could be this question asked find least learned or cost or something ,where define a local variable in that method of class type with null and then compare and find lowest value
"The toString method should output the entire dictionary content including the knowledge values (but of course without the null references), one entry per line. -- technicaldic class has entries array , so display all of it in for loop public class TechnicalDictionary { private Entry[] entries; @Override public String toString() { StringBuilder sb = new StringBuilder(); for (Entry entry : entries) { if (entry != null) { sb.append(entry.toString()).append(""\n""); } } return sb.toString(); }"
A student is generated by a constructor that expects the same parameters as the constructor for instances of type UniMember ,meaning --means same param in constructor like this public Student(String firstName, String lastName, String address) {
Professor additionally have an array assistants, in which assistants are stored. A professor can have a maximum of 10 employees. private Assistant[] assistants; , in constructor this.assistants = new Assistant[10];
"if (assistantCount < 10) { assistants[assistantCount++] = assistant; } else { System.out.println(""Maximum number of assistants reached!""); System.exit(0); } ###way to add new elem with incremental counter for index of array"
You deleted this message
"sb.append(super.toString()).append("", Assistants: ""); --first inserting superclass content and then in for loop for (int i = 0; i < assistantCount; i++) {
sb.append(""\n\t"").append(assistants[i].toString()); }-- add assistant class content for each eleme of assistant"
in constructor, sometimes we have to do this
supervisor.addAssistant(this);
After a x = q.deQueue() you must first determine to which of the three classes x actually belongs --- if (x instanceof String) { stringLengthSum += ((String) x).length(); } else if (x instanceof Integer) { integerSum += (Integer) x; } else if (x instanceof Boolean) { if ((Boolean) x) { booleanTrueCount++; } } ##nice conc
"Save the read data in a ArrayList<Product> file -- products.add(new Product(line)); Write a constructor for Product that appropriately splits a read line (i.e. a string) into these three parts. artno and name should be stored without the enclosing quotation marks this.price = Integer.parseInt(part.split("":"")[1].trim()); for (Product product : products) { bw.write(product.toString()); bw.newLine(); } ##each product contains 3 variable in order , being asked"

int averagePrice = (int) Math.floor((double) total / products.size());-
to display enter it (rounded down to an integer)
"public Product(String line) { String[] parts = line.split(""",\\s*"");
for (String part : parts) { if (part.startsWith(""artno:"")) { this.artno
= part.split("":"")[1].replaceAll(""\"""", """).trim(); } else if
(part.startsWith(""name:"")) { this.name =
part.split("":"")[1].replaceAll(""\"""", """).trim(); } else if
(part.startsWith(""price:"")) { this.price =
Integer.parseInt(part.split("":"")[1].trim()); } } }"
".replaceAll(""\"""", """) What It Does: Replaces all double-quote
characters ("") in the string with an empty string, effectively removing
them"
Why Use replaceAll()? It's a method for replacing all occurrences of a
pattern (using regular expressions).
"replace() does a literal character sequence replacement. replaceAll()
treats the first argument as a regular expression patternString result =
text.replace(""\"""", """);"
array[i] = Integer.parseInt(tokens[i + 1]); -- storig string array into
int array
boolean[] visited = new boolean[size]; visited[index] = true , if
(visited[index]) --conceptual for when storing if visited or not , or
happened or not
while (steps < 15) { index = array[index]; steps++;}
boolean[] isPrime = new boolean[m + 1]; if (isPrime[i]) { primeCount++;
}--storing if something is true or not and then using that info to get
total count
for (int i = 2; i * i <= n; i++) {-- for if prime or not
class SpaceAsset { protected double[] coordinates = new double[3];}
"return ""Coordinates: ("" + coordinates[0] + "", "" + coordinates[1] +
"", "" + coordinates[2] + ""), Speed: "" + speed;"
coordinates is protected , so can be accessed in subclass to set with new
variable
public void move(double x, double y, double z) { coordinates[0] = x;
coordinates[1] = y; coordinates[2] = z; }
"for (int i = n; i <= m; i++) { if (i % k == 0) { result.add(""beep""); }
else { result.add(String.valueOf(i)); } ##storing integer and string
value in list of string"
while ((long) n * n <= maxInt) { n++; } --when checking boundayry of one
type of variable , casting into another variable, since finding min n
return reverse(string.substring(1)) + string.charAt(0);
another way to count no of unique words in string // Create a set to
store unique words Set<String> uniqueWords = new HashSet<>(); // Add
each word to the set (duplicates will be ignored) for (String word :
words) { uniqueWords.add(word); } store in set and get size of set
make rear null when front null, when both are null ,queue is empty
rear.setNext(newElem);front = front.getNext();
abstract class SampleSet { // Array to store the samples protected
Sample[] samples;}
when we have array of fixed size in class, suppose of 100 , initilaize
one counter in constructor which tracks index of array when filling new
value , so when new elem to add , use this variable to store in first
empty pos
public int getNumberOfWords() { return wordCount; } ### returning size of
array ,not original size but till we have some elem in array

for true loop to come out , need to use break ;
"// Create the dot string String dots = "".repeat(numDots); --creating string with same string , numDots is integer value"
"return String.format(""%02d:%02d:%02d"", hours, minutes, seconds);-- when returning string with desired format"
using stringbuilder to add into one string using 2 logic or in 2 parts
when one class contains array of variables or array of another class then ,its initilization is ArrayDictionary dictionary = new ArrayDictionary(5);
// Count repeated characters if (i + 1 < s.length() && s.charAt(i) == s.charAt(i + 1)) { count++;
"when to give Input: ""aabccccccaaa"" Output: ""a2b1c5a3"""
"Input: [""eat"", ""tea"", ""tan"", ""ate"", ""nat"", ""bat""] Output: [[""eat"",""tea"",""ate""], [""tan"",""nat""], [""bat""]] when putting similar kind of elem in one , use concept of map with same key"
map.putIfAbsent(sorted, new ArrayList<>()); map.get(sorted).add(s);
String Searching and Matching (Regex) Concepts Covered: Finding Substrings (indexOf, lastIndexOf) Pattern Matching (matches, contains) Regular Expressions (Pattern, Matcher)
"long startTime = System.currentTimeMillis();   long endTime = System.currentTimeMillis(); System.out.println(""Time with String: "" + (endTime - startTime) + "" ms"");"
int rows1 = mat1.length; int cols1 = mat1[0].length; int cols2 = mat2[0].length; for (int i = 0; i < rows1; i++) { for (int j = 0; j < cols2; j++) { for (int k = 0; k < cols1; k++) { result[i][j] += mat1[i][k] * mat2[k][j];
rotated[j][n - 1 - i] = mat[i][j]; -90 degree clockwise rotation matrix
return rotate90(rotate90(mat)); --// Rotate 180 degrees
angel rotation , one over other   return rotate90(rotate180(mat));
for (int[] interval : merged) {
System.out.println(Arrays.toString(interval));
using 2 sets, when elemet could not be added in set1 means its already there , add in diff set
"BufferedReader br = new BufferedReader(new FileReader(""sample.txt""));
It will work as long as the sample.txt file is located in the current working directory where the program is being run. This is because ""sample.txt"" is treated as a relative file path, and Java will try to locate it in the current directory of the project or execution context."
"File file = new File(""sample.txt""); -- when file is not in same directory"
"String[] words = line.split(""\\W+""); // Split based on non-word characters"
try (BufferedWriter writer = new BufferedWriter(new FileWriter(file, true))) when appending
try (BufferedReader br = new BufferedReader(new FileReader(inputFile)); BufferedWriter bw = new BufferedWriter(new FileWriter(outputFile)))
"interface Printer { default void operate() { System.out.println(""Printing a document.""); } class MultiFunctionDevice implements Printer, Scanner { @Override public void operate() { Printer.super.operate();"
interface Container<T>
return str.contains(String.valueOf(sum));-- checking sum which was int , into string like a13bc9dgc, like 13 in this string

"Student s1 = new Student(""1234;John Doe;85""); --this is passed as string , need to split and give result"
when said
Tuple<Integer, Integer> t1 = new Tuple<>(4, 2);
this.numberOfGenres = genres.size();- another way to fill one variable
if a[i] is not valid (beacuse a[a[i]] cannot be computed) - when goes beyon index , in this case a[i] >= a.length
a string is passed Pizza Hawaii 10.50, create object (recipe,price) split with space and then first 2 into recipe and last one into price , but first 2 can be converted into one string either by using string,join() or in stringbuilder
"String d=String.join("" "",s2[0],s2[1]); System.out.println(d);"
"String[] words = {""Java"", ""is"", ""fun""}; // Joining words using a space as a delimiter String result = String.join("" "", words); System.out.println(result); // Output: Java is fun"
Calculate the length of each part. The length of each part will be length of string / n
parts[i] = str.substring(i * partLength, (i + 1) * partLength);
if (index > str.length() - 3) { return 0; }
// Method to get the list as an int[] public int[] toIntArray() { int[] arr = new int[numberList.size()]; for (int i = 0; i < numberList.size(); i++) { arr[i] = numberList.get(i); } return arr; }
"if (tag.startsWith(""#"")) { tagsList.add(tag); }- way to check extra condition before adding into diff collection"
storing date in private LocalDate purchaseDate;
LocalDate.of(2025, 2, 16)
Parallelogram pg = new Parallelogram(a,b,c); --pg is object , parallelogram is reference type
(Object)value).getClass().getSimpleName()
Quadrilateral qlRef = new Parallelogram(a,b,c); if (qlRef instanceof Parallelogram) pgRef = (Parallelogram) qlRef;}
Parallelogram pgRef = new Parallelogram();
Quadrilateral qlRef = new Parallelogram(a, b, c); Here, qlRef is a reference of type Quadrilateral, but it's pointing to an object of type Parallelogram
pgRef = (Parallelogram) qlRef; performs downcasting, converting the reference type from Quadrilateral to Parallelogram. Now, pgRef and qlRef both point to the same object, but pgRef can access methods and fields specific to Parallelogram
linked list push -push newElem.setNext(top);, top = newElem; Elem temp = top; top = top.getNext(); return temp.getObject();
Integer iObj = new Integer (i);
int num = (int) (Math.random() * 2); Multiplying by 2 gives a range from 0.0 to 1.999.... Casting to int truncates the decimal, resulting in either 0 or 1.
double num = 4.5; long roundedValue = Math.round(num); // Output: 5
Math.ceil() and Math.floor() Math.ceil() rounds up to the nearest integer. Math.floor() rounds down to the nearest integer.
while (fast != null && fast.next != null) { slow = slow.next; fast = fast.next.next; } ##using 2 pointer to find mid of elem
to string ##very imp for (int i = 1; i < s.length(); i++) { if (s.charAt(i) == s.charAt(i - 1)) { count++; } else { compressed.append(s.charAt(i - 1)).append(count); count = 1; } }
compressed.append(s.charAt(s.length() - 1)).append(count);

StringBuilder has an append(int i) method. This method converts the integer to its string representation and appends it to the StringBuilder.
In Java, a String is immutable, which means once it's created, it cannot be changed. This is why you can't directly sort the characters in a String. Sorting requires rearranging characters, which involves modifying the order of elements
arrays are mutable in Java
class ClassName<T> { // T is a type parameter private T data; public ClassName(T data) { this.data = data; } public T getData() { return data; } }
"Pair<String, Integer> pair = new Pair<>(""Age"", 30);
System.out.println(pair.getKey() + "": "" + pair.getValue()); , custom collection using generic class concept"
class NumberBox<T extends Number> {   <T extends Number> ensures T is a subclass of Number, allowing intBox and doubleBox but not stringBox.
"public static <T> void printArray(T[] array) { for (T element : array) { System.out.print(element + "" ""); } System.out.println(); }"
interface Pair<K, V> { K getKey(); V getValue(); } class KeyValue<K, V> implements Pair<K, V> {
"Pair<String, Integer> pair = new KeyValue<>(""Age"", 30);"
int num = ch - '0'; // Convert char digit to int char letter = (char) (num + 96); // Get corresponding letter
while (matcher.find()) { result.append(matcher.group()); }
"String regex = ""a?b""; // Match ""b"" or ""ab"""
"String regex = ""(abc)+""; , abc is altogether"
"System.out.println(""abcd"".matches(regex)); // false"
for (int i = 1; i < chArray.length; i++){ 13 for (int j = 0; j < chArray.length - i; j++) 14 if (chFeld [j] > chArray [j+1]) { 15 char help = chArray [j]; 16 chArray [j] = chArray [j+1]; 17 chArray [j+1] = help; 18 }
getKey(pos.getNext().getObject()) > key)
pos.setNext(newElem); end = newElem; 2 newElem.setNext(null);
newElem.setPrev(pos);
With ‘>>’ the sign is preserved, With ‘>>>’ negative numbers turn positive
"public NotComparableException(Pet pet) { if (pet instanceof Cat) { this.message = ""Exception: Cannot compare cats to other species""; } else if (pet instanceof Dog) { this.message = ""Exception: Cannot compare dogs to other species""; } else { this.message = ""Exception: No comparison possible""; } } public String getErrMsg() { return this.message; }"
dictionary[i] = new DictionaryElement(key, value);
if(records.containsKey(name)){ if(time< records.get(name)){ records.put(name, time); } } else{ records.put(name, time); }
"try{ id = Integer.parseInt(parts[0]); } catch (Exception e) { throw new IllegalArgumentException(""Invalid ID format. ID must be an integer.""); }"
In this case, the catch block catches a generic Exception from Integer.parseInt() (like NumberFormatException) and re-throws it as an IllegalArgumentException with a clearer message. ##imp
"if (operator.equals(""">"")) { if (left > right) { return ""valid"";"
Each object (slr1, slr2, and phone) has its own pictures
"System.out.println(""\""""");-"""

"// Optional: Check if the new index is out of bounds if (index < 0 || index >= size) { System.out.println(""Index out of bounds. Ending the hunt.""); break; }"
"return ""\""Invalid Sequence\""""";--to print in quote too"
String s1 = s.substring(0,mid); String s2 = s.substring(mid);
int[] res = new int[Math.min(a.length, b.length)];
sb.append(String.valueOf(shiftchr));
No, you don't have to explicitly convert a char to a String before appending it to a StringBuilder
insert() Inserts data at a specified index ##very imp to use when want to have string in specific order
delete() and deleteCharAt() delete(start, end): Removes a substring from start to end-1. deleteCharAt(index): Removes a character at the specified index
"StringBuilder sb = new StringBuilder(""Hello World""); sb.replace(6, 11, ""Java"");"
sb.reverse();
stringbuilder -indexOf() and lastIndexOf(), length() and capacity() length(): Returns the number of characters. capacity(): Returns the current capacity (increased automatically as needed).
String sub = sb.substring(6); // From index 6 to end
char ch = sb.charAt(1); // 'e' sb.setCharAt(1, 'a');
StringBuilder: Not thread-safe. Multiple threads can access it simultaneously without synchronization, leading to potential data inconsistency. StringBuffer: Thread-safe. It is synchronized, making it safe for use in multithreaded environments
int numberOfSentences = sc.nextInt(); sc.nextLine();
if (parts[i].equals(parts[j])) break;
List<String> tagList = new ArrayList<>(Arrays.asList(tags));
calling this   String var=post123.toString().toLowerCase(); to store one class all attribute into string using tostring function
public SimplePlaylist(int capacity) { songs = new Song[capacity]; count =0; }-- for every object also have count value which has current filled index value of array
"public Employee() { super(); // Calls the default constructor of Person this.role = ""Not Assigned""; }"
"public Person() { this.name = ""Unknown""; this.age = 0; }"
"""artno: \""""" + artno + ""\"""
bw.write(product.toString());
"""artno: \""""" + artno + ""\"", name: \""""- putting quote in value of variable like ""A1234"" artno: ""A1234"","
"if asked to write in this form, artno: ""A1234"", name: ""Laptop"", price: 1200 , all stuff except variable value should inside quote"
"including , :"""
File f=new File(filename);   if(f.exists())
for(int i=0;i<100;i++) { id[i]=null; pw[i]=null; } ####initializing null
// Constructor to initialize with an Employee object public EmployeeManager(String name, String designation, int age) { this.employee = new Employee(name, designation, age); }
sc.next()-Use When: You want to read a single word or a single token
sc.nextLine(): Purpose: Reads the entire line of input until the end of the line. Returns: A String containing the entire line.
int age = sc.nextInt(); sc.nextLine(); // Consume the newline String name = sc.nextLine();

After using nextInt(), nextDouble(), or next(), add an extra sc.nextLine() to consume the leftover newline
"int age = sc.nextInt(); sc.nextLine(); // Consume the leftover newline System.out.print(""Enter your name: ""); String name = sc.nextLine();"
private ArrayList<Employee> employees;
public void addProject(Project project) { if (projects == null) { projects = new ArrayList<>(); // Ensure projects is initialized } ### one way too
Check for Common Mistakes First Look for: Off-by-one errors in loops (i < n vs. i <= n) Array index out of bounds (index >= array.length) NullPointerException if using objects without initializing them Infinite loops (while(true) without a break condition) Logical errors in conditions (if (a = b) instead of if (a == b))
Track loop counters (i, j) and array indices.
C[i][j] += A[i][k] * B[k][j];--matrix multiplication
Fix: Check if the key already exists before updating.
public boolean equals(Object o) { if (this == o) return true; if (o == null || getClass() != o.getClass()) return false; Person person = (Person) o; return Objects.equals(name, person.name); } ##comparing object .##imp this to object
Fix: Use LinkedHashSet or TreeSet for predictable order. Set<String> set = new LinkedHashSet<>();
Mistake: Using null elements in TreeSet, which relies on sorting . Fix: Avoid nulls or use HashSet, which allows null values imp
"Set<String> set = new HashSet<>(Set.of(""A"", ""B""));"
"List<String> list = Arrays.asList(""A"", ""B""); list.add(""C""); // Error: UnsupportedOperationException List<String> list = new ArrayList<>(Arrays.asList(""A"", ""B"")); list.add(""C""); // Works fine"
poll() in Queue: Removes and returns the head (front) of the queue.
public <T> ReturnType methodName(T param) { // Method body }
"public static <T> void printArray(T[] array) { for (T element : array) { System.out.print(element + "" ""); } System.out.println(); }"
public static <T, U> boolean compare(T first, U second) { return first.equals(second); }
"public class GenericCompare { // Generic method with two type parameters public static <T, U> boolean compare(T first, U second) { return first.equals(second); } public static void main(String[] args) { // Calling with Integer and String boolean result1 = compare(10, ""10""); System.out.println(result1); // Output: false // Calling with String and String boolean result2 = compare(""Hello"", ""Hello""); System.out.println(result2); // Output: true } }"
Using a bounded type to restrict the type parameter to Number and its subclasses.
public static <T extends Number> double calculateSum(T num1, T num2) { return num1.doubleValue() + num2.doubleValue(); }
"public static <T> void display(T item) { - this is method , u can use it to print any kind of datatype like Utility.display(123);   // Output: Item: 123 Utility.display(""Hello"");   // Output: Item: Hello Utility.display(45.67);"
public static <T> void printList(List<T> list) {
public <T> T methodName(T param) {
public static <T extends Comparable<T>> T findMax(T a, T b) { return (a.compareTo(b) > 0) ? a : b; }

<T extends Class> to restrict the return type to subclasses of a certain class.
You deleted this message
Use instanceof to check the type before casting. if (p instanceof Child) { Child c = (Child) p; }
whenever some exception is thrown and not handled, catch it
IOException when handling file
"public void exampleMethod() { Exception e = new Exception(""Error message""); throw e; } or public void exampleMethod() { throw new Exception(""Error message""); }"
The method signature in the implementing class must match exactly with the one in the interface.
"method in interface default void draw() { // Default method System.out.println(""Drawing..."""); }"
public void show() { // Resolving conflict A.super.show(); // OR B.super.show(); A is interface }
while (temp.next.next != null) { --sometimes can increment by 2 next
Incorrectly downcasting without checking the instance type. Parent p = new Parent(); Child c = (Child) p; // Runtime error: ClassCastException
✓ Fix: Always check using instanceof before downcasting. if (p instanceof Child) { Child c = (Child) p; }
Primitive to Object Type Casting (Autoboxing & Unboxing) Autoboxing (Implicit) ✓ Correct Example: int num = 5; Integer obj = num; // Autoboxing (int to Integer) Unboxing (Explicit, but Automatic) ✗ Common Error: Trying to cast an Integer to int incorrectly. Integer obj = null; int num = obj; // NullPointerException ✓ Fix: Always check for null before unboxing. if (obj != null) { int num = obj; }
"Object obj = ""10""; int num = Integer.parseInt((String) obj); // Correct way System.out.println(num); // Output: 10"
"Object obj = 10; int num = (int) obj; // Works fine if obj is Integer Object obj2 = ""10""; int num2 = (int) obj2; // ERROR: ClassCastException"
Object obj = 100; String str = String.valueOf(obj); // Safe conversion
"Object obj = ""Hello""; String str = (String) obj; // Works fine"
Fix: Use Modulo to Loop Within Range char c = 'Z'; c = (char) ((c - 'A' + 1) % 26 + 'A'); System.out.println(c); // Output: A ##imp
"##very imp char shifted = (char) ((ch - 'a' + shift) % 26 + 'a'); when shifting each char of ""shjtzyw"" by 5 but in range of a-z ascii value of a 97, z 1222 so ch-a give relative positon , then shift it and divide by 26 to get how many position it has to go beyond a if diff is more than 26 between ch-a+shift"
char c = '5'; int num = Character.getNumericValue(c);
System.out.println(num); // Output: 5 or int num = c - '0';
System.out.println(num); // Output: 5
"converting char to string String.valueOf(c) + ""B"" and then concaenating"
"String s = null; String result = s + ""Hello"";
System.out.println(result); // Output: nullHello (Unexpected) String s = null; String result = (s != null ? s : """") + ""Hello"";
System.out.println(result); // Output: Hello"
if (s.isEmpty()) {
Fix: Use StringBuilder for Efficient Concatenation

"String result = input.replace(""[0-9]"", """"); // Incorrect String result = input.replaceAll(""[0-9]"", """");"
System.out.println(Arrays.toString(charArray)); // Output: [H, e, l, l, o]
// Output: [H, e, l, l, o]
❌ Common Error: Using (String) char char ch = 'J'; String str = (String) ch; // Error: Cannot cast char to String char ch = 'J'; String str = String.valueOf(ch); // Correct way System.out.println(str);
"String regex = ""\\d*[.]""; // ✅ If you want an actual dot"
"""\\d*[.]"" ""123."", ""."" Matches digits followed by any character (since . is unescaped). ""\\d*\\."" ""123."", ""."" Matches digits followed by a literal dot (.)."
"String regex = ""\\d*[.]""; (Correct, matches only a literal dot)
Correct if you want to match a dot literally. 🔹 The [.] means ""match exactly one dot (.)"". Since [] defines a character set, the . inside [] loses its special meaning ""123.""✅ Yes"
"Pattern pattern = Pattern.compile(""\\d+""); Matcher matcher = pattern.matcher(""abc123""); if (matcher.matches()) { // ❌ Won't work since `matches()` checks the whole string System.out.println(""Match found""); }"
"if (matcher.find()) { // ✅ Use `find()` to check for partial matches System.out.println(""Match found""); } ##when doing partial matching"
"String[] fruits = input.split(""[,;\\s]+""); // ✅ Correct: Spaces included"
// Separate arrays for different character sets int[] lowercase = new int[26]; int[] uppercase = new int[26]; int[] digits = new int[10]; if (Character.isLowerCase(ch)) lowercase[ch - 'a']++; else if (Character.isUpperCase(ch)) uppercase[ch - 'A']++; else if (Character.isDigit(ch)) digits[ch - '0']++;
freq[(int) 'a' + 1]++; // ✅ Now it's clear we're tracking 'b'
static void show() { System.out.println(this.x); // ❌ Compilation Error: Cannot use 'this' in a static context }
❌ Compilation Error: Cannot override static method
A static method does not have access to this (instance context) because it belongs to the class, not an object. Since non-static members require an instance, a static method cannot directly access them.
see linkedlist queue and array problem on exam day
matcher.find() Searches for any substring within the input string that matches the rege
matcher.matches() Checks if the entire input string matches the regex pattern
"// Checking if ""Java"" is present at index 18 boolean match = text.regionMatches(18, keyword, 0, keyword.length());"
"example.add(index + 1, ""star""); -inserting after specific point"
"""\nStreet:"" This moves the output to a new line before printing ""Street:"""
if(digit>max) { max=digit; } ## finding largest digit
when sometimes , i asked to find continous match in string , put in if condiiton if it not matches then break else add in strinbuilder
// Loop through and add matching symbols to StringBuilder while (matcher.find()) { symbols.append(matcher.group()); }

char[] rearranged = new char[word.length()]; return new String(rearranged);
A Human has-a father (who is a Man). A Human has-a mother (who is a Woman). This creates object relationships where a human is composed of other human objects.
Concept: Instead of inheritance, composition allows objects to contain other objects as instance variables.
finding loop   for (int i = 0; i < arr.length; i++) { if (!visited[i]) { int current = i; while (!visited[current]) { visited[current] = true; current = arr[current]; } loopCount++; // A loop is completed } }
rand.nextInt(20) generates a random integer from 0 to 19 (inclusive).
int num = (rand.nextInt(11) - 5) * 2; --Generate a random even number between -10 and 10.
In Java, break only exits the loop in which it is directly placed. It does not affect any outer loops.
when we have while (true) {, there would be some if condition if fulfilled, it breaks and end loop
also in loop put this condition or in general when index=arr[index] if (current < 0 || current >= arr.length) { break;
##very imp
when asked to find distinct loop, use one visited[] array to track if visited or not, if asked of no of loop , just check when visited[] is true and then increment count while (true) { if(visited[index]) { count++; break; } visited[index] = true; index = a[index]; if (index < 0 || index >= a.length) { break; // break; } if(!visited[i]) { int index = i; while (!visited[index]) { visited[index] = true; index = a[index]; // break; } count++; }
u return in method and break in normal loop
use HashSet to find something repeating like first repeating number or char . loop through set and once it has , break and return else feed elem into set
fast = arr[arr[fast]];- could be used to solve some problem
longestCycle = Math.max(longestCycle, cycleLength);
also you can use length counter to keep track of no of elem u are traversing to
detectSelfLoops when a[i]==i
also can use list to store elem of loop while traversing
reversed = reversed * 10 + digit;
If either operand is a String, it converts the other operand to a String and concatenates them
use stringbuilder when want content from diff string or list . also in diff order
private Assistant[] assistants = new Assistant[10];
public class Professor extends Staff { private Assistant[] assistants = new Assistant[10]; private int assistantCount = 0; public void addAssistant(Assistant assistant) { if (assistantCount < 10) { assistants[assistantCount++] = assistant; } }## each assistantCount is counter for array , for every objec
if (supervisor.assistants[i] != this) { ##- one way to delete some object from list. this keyword is used to add or delete object
import package1.OuterClass -Accessing Inner Class from Another Class in a Different Package
package package2;

static String[] splitAddress(AddressOptimizer ao), when calling this method, we pass this keyword as args String[] parts = AddressUtil.splitAddress(this);
method called in AddressOptimizer class so object is of AddressOptimizer type in method
always in method , handle condition which ask u send null or return or make it false and then work on updating or manipulating data
in method with some return type like int or boolean , return (don't use break), in method with void return type , just call return so that if condiiton need to be handled ,is handeled
you can also use set or linkedlist to store info about whether this indices is has been visited or not by checking data in this set/list
visited.add(currentPos); by if (visited.contains(currentPos)) {
long startTime = System.nanoTime();
if (s.charAt(i) == s.charAt(i + 1)) { pairCount++; i++; // Skip the next character as we've already counted this pair }
this.coordinates[0] = x;
double[] newCoordinates = {x, y, z};
super.getCoordinates()[0] = newCoordinates[0]; --updating new coordinates
comparing and casting into that datatype if (x instanceof String) { // If it's a String, add the length to totalStringLength totalStringLength += ((String) x).length(); }
when writing to file from class object, you have to call tostring method when object could be of list tape, so looping through object and writing each to file
convert string to double or int when reading a string line and parsing it
// Swap the two elements in the linked list x.setNext(second);
// The next of 'x' now points to the second element
first.setNext(second.getNext()); // The first element now points to the third element second.setNext(first);   // The second element now points to the first element
in array object of class, its good to check if its a[i] is null or not before starting any operation or any changes
when use mapputifabsent(genre,new arraylist) , next line could be map.get(genre).add(movies)
for question like common prefix, add character in some stringbuilder till there is no match and once this condition fail, break out
when asked to return object, u can return null as default
If found, remove that character from both strings and recurse with the shortened strings. this concept can be used in some way
use of index can also be used to find char or string in another string
// Take the first character of s1 char ch = s1.charAt(0); // Find its position in s2 int index = s2.indexOf(ch); String newS1 = s1.substring(1); // Remove first character from s1 String newS2 = s2.substring(0, index) + s2.substring(index + 1);
after finding pos of char, you can use substring to delete at char in a way , not considering it further
"String sentence = ""good morning, goodluck with the goal""; String pattern = ""\\bg\\w*""; // Words starting with 'g'"
"String pattern = ""g\\w*""; g: Matches the character 'g'."
"String vowelPattern = ""[aeiouAEIOU]""; // Pattern for consonants String consonantPattern = ""[a-zA-Z&&[^aeiouAEIOU]]"";"
else if (ch >= 'a' && ch <= 'z') {

order of if and else condition , the shorter one is in if and rest one in consonant
// Convert to uppercase by subtracting 32 ch = (char) (ch - 32);
if (ch == oldChar) { result.append(newChar); , result is stringbuilder
Using Stack to reverse the order of characters.
while (!stack.isEmpty()) { reversed.append(stack.pop()); }
checking if parenthesis is balanced or not , firs put opening parenthesi in stack and then compare after pop first one and checking it should match with the closed one
char open = stack.pop(); if ((ch == ')' && open != '(') ||
chekcing stack.peek() <= arr[i] ,
TreeMap<String, Integer> sortedMap = new TreeMap<>(map);-- ordering elem by sorting
// Sort the ArrayList Collections.sort(numbers);
stack implementation of queue // Enqueue operation public void enqueue(int data) { stack1.push(data); } in dequeue , use stack.pop
while (fast != null && fast.next != null) {- for finding middle elem
in writing to file , when iterate through list of array of object like product array, in writer.write(content) content should be according to what asked, can call method of class like tostring or get and then put in file likewise , if special data after fulfilling some codiition , then check it
IOException,NoSuchFileException
Files.createFile(filePath);, (Files.exists(filePath)) for using Files.write() Collect the data first (e.g., using a List<String> names) into list , then write all at once using Files.write(filePath, names);
Files.write(filePath,
List.of(person.getName())StandardOpenOption.APPEND);
"List<String> fruits = Arrays.asList(""Apple"", ""Banana"", ""Cherry"");
fruits.set(1, ""Blueberry""); // Allowed System.out.println(fruits); // Output: [Apple, Blueberry, Cherry] When you need a modifiable list but with a fixed size. When you want to update existing elements."
"List<String> fruits = List.of(""Apple"", ""Banana"", ""Cherry""); When you need an unmodifiable list."
an enum is a special data type representing a group of constants
enum values are implicitly public static final, making them constants.
"private Map<T, U> map;   this.list = new ArrayList<>(); this.map = new HashMap<>(); even when its generic type System.out.println(""Type of T (List): "" + list.get(0).getClass().getName()); Container<String, Double> stringDoubleContainer = new Container<>();-- this is how u declare object with datatype for class ""class Container<T, U>"" ##imp"
Integer.parseInt(digits.toString()))   where digit is stringbuilder
// Check for overflow/underflow before appending digit if (reversed > Integer.MAX_VALUE / 10 || reversed < Integer.MIN_VALUE / 10) { return 0; }
An Armstrong number of n digits is equal to the sum of its own digits
get the length of digit by converting into string int digits = String.valueOf(num).length();
Inside square brackets, most special characters lose their special meaning. This includes the dot (.), which is treated as a literal dot.
[A-Za-z0-9.-]
interface Container<T> { class MyContainer<T> implements Container<T> {
, generic interface

List<?> can accept any type of List, whether it's List<String>, List<Integer>, or anything else.
Object[] objArray = new Object[3]; // Autoboxing: Storing primitives as Object objArray[0] = 100;   // int -> Integer -> Object objArray[1] = 99.99;   // double -> Double -> Object objArray[2] = 'A';   // char -> Character -> Object // Retrieving with type casting int intValue = (int) objArray[0]; double doubleValue = (double) objArray[1]; char charValue = (char) objArray[2];
"Object obj = 12345; String str = obj.toString(); if (obj instanceof String) { System.out.println(""String: "" + obj); }"
"""Class Type: "" + obj.getClass().getSimpleName()"
if (obj instanceof String) { String str = (String) obj; , check and then do type casting
"Object obj1 = ""Hello""; Object obj2 = ""Hello""; Object obj3 = new String(""Hello""); (obj1 == obj2)); // true (obj1 == obj3)); // false obj1.equals(obj2)); // true obj1.equals(obj3)); // true"
String input = sc.next(); // Read as String first char ch = input.charAt(0); // Extract the first character ##way to read character from input
sc.next() returns a String
TrafficLight light = TrafficLight.valueOf(input); Converts a String to an Enum Constant: This line converts the String value stored in input into the corresponding TrafficLight enum constant.
It takes a String argument, which should exactly match the name of one of the enum constants (case-sensitive). If a match is found, it returns the corresponding enum constant.
"if (Math.abs(ch1 - ch2) == 1) { System.out.println(""The characters are consecutive.""); --this one , can be used"
Integer to Character: num + '0'
Integer.compare(num1, num2)-use of these 2 compare between 2 variables
Integer wrappedInt = primitiveInt; int unboxedInt = wrappedInt.intValue(); like this for another datatype
// Final variable (constant) final int MAX_VALUE = 100;
calling a static method using an object reference (like obj1.displayCount()) is not the recommended practice, though it does work without causing a compilation error.
ava allows calling a static method using an object reference, but under the hood, the method is resolved at the class level. This means:
obj1.displayCount(); is actually interpreted as:
StaticExample.displayCount();
public class UseFinalClass { // Composition: Including FinalClassExample as a member private FinalClassExample finalObj; FinalClassExample is final class
UtilityClass.displayInfo();- directly calling method using class name in which it was defined
public class UtilityClass { public static void displayInfo() {
"if (""AEIOUaeiou"".indexOf(ch) == -1) { --- when i find which is inverse of given condition"
while ((index = sb.indexOf(target)) != -1) {-You need the index value later in the loop ,like the line in stringbuffer reader
It is inefficient if indexOf(target) is called multiple times inside the loop because the same search operation is repeated
Find First Non-Repeating Character- putting count freq in map and then in sam eloop checking if its count is 1 means it has come more than 1 times

```
"colors.set(1, ""Yellow""); // Changes ""Green"" to ""Yellow""
colors.remove(0); // Removes ""Red"""
"Queue<Integer> queue = new LinkedList<>(); queue.add(""Apple"");  for
(String item : queue) {"
// Push all elements to stack while (!queue.isEmpty()) {
stack.push(queue.remove()); }
LinkedList<Integer> list = new LinkedList<>();
interface Pet extends Animal { void play(); }
Has a static variable count to keep track of the number of objects
created.
List<List<Integer>> listOfLists = new ArrayList<>(numberOfRows); for (int
i = 0; i < numberOfRows; i++) { listOfLists.add(new ArrayList<>());  //
Initializing each inner list }
```