**1. Write a for loop that prints the even numbers from 1 to 20.**

java

Copy code

public class Main {

public static void main(String[] args) { for (int i = 1; i <= 20; i++) {

if (i % 2 == 0) {

System.out.println(i);

}

}

}

}

**2. Create a while loop that prompts the user for their flight choice until a valid number is entered.**

java

Copy code import java.util.Scanner;

public class Main {

public static void main(String[] args) { Scanner scanner = new Scanner(System.in); int flightChoice; while (true) {

System.out.print("Enter your flight choice (positive number): "); if (scanner.hasNextInt()) { flightChoice = scanner.nextInt(); if (flightChoice > 0) {

System.out.println("Valid flight choice: " + flightChoice); break;

}

} else {

scanner.next(); // clear invalid input

}

System.out.println("Invalid input. Please enter a valid number.");

}

}

}

1. **Discuss the pros and cons of using different types of loops for iterating through an array of numbers.**

**Pros and Cons:**

* + **For Loop:**
    - Pros: Suitable when the number of iterations is known. Provides better control with initialization and increment steps. o Cons: Less intuitive for scenarios with unknown bounds or dynamic conditions.
  + **While Loop:**
    - Pros: Flexible for cases where the number of iterations is unknown. Best for conditions checked before the first iteration.
    - Cons: Risk of infinite loops if conditions are not handled properly.
  + **Do-While Loop:**
    - Pros: Ensures the code block executes at least once, regardless of the condition.
    - Cons: Less commonly used, which might make code less readable.

1. **Write a Java program that uses a for loop to print the first 10 numbers of the Fibonacci sequence.** java

Copy code public class Main { public static void main(String[] args) { int n1 = 0, n2 = 1;

System.out.print("First 10 Fibonacci numbers: " + n1 + " " + n2);

for (int i = 2; i < 10; i++) { int next = n1 + n2;

System.out.print(" " + next); n1 = n2; n2 = next;

}

}

}

**5. Create a Java program using a while loop to calculate the sum of integers from 1 to 100.** java

Copy code public class Main { public static void main(String[] args) { int sum = 0, i = 1; while (i <= 100) { sum += i; i++;

}

System.out.println("Sum of integers from 1 to 100: " + sum);

}

}

**6. Implement a do-while loop that prompts the user to enter a number until they enter a negative number.**

java

Copy code import java.util.Scanner;

public class Main {

public static void main(String[] args) { Scanner scanner = new Scanner(System.in); int number;

do {

System.out.print("Enter a number: "); number = scanner.nextInt();

} while (number >= 0);

System.out.println("You entered a negative number. Exiting.");

}

}

**7. Write a Java program that demonstrates the use of the continue statement in a loop.** java

Copy code public class Main { public static void main(String[] args) { for (int i = 1; i <= 10; i++) { if (i % 2 == 0) {

continue; // Skip even numbers

}

System.out.println("Odd number: " + i);

}

}

}

**8. Initialize and print a 2D array of integers in Java.** java

Copy code public class Main { public static void main(String[] args) {

int[][] array = { {1, 2, 3},

{4, 5, 6},

{7, 8, 9}

};

for (int i = 0; i < array.length; i++) { for (int j = 0; j < array[i].length; j++) {

System.out.print(array[i][j] + " ");

}

System.out.println();

}

}

}

Continuing with solutions for questions 6 to 64:

**6. Implement a do-while loop that prompts the user to enter a number until they enter a negative number.**

java

Copy code import java.util.Scanner;

public class Main {

public static void main(String[] args) { Scanner scanner = new Scanner(System.in); int number;

do {

System.out.print("Enter a number: "); number = scanner.nextInt();

} while (number >= 0);

System.out.println("You entered a negative number. Exiting.");

}

}

**7. Write a Java program that demonstrates the use of the continue statement in a loop.** java

Copy code public class Main { public static void main(String[] args) { for (int i = 1; i <= 10; i++) { if (i % 2 == 0) {

continue; // Skip even numbers

}

System.out.println("Odd number: " + i);

}

}

}

**8. Initialize and print a 2D array of integers in Java.** java

Copy code public class Main { public static void main(String[] args) {

int[][] array = {

{1, 2, 3},

{4, 5, 6},

{7, 8, 9}

};

for (int i = 0; i < array.length; i++) { for (int j = 0; j < array[i].length; j++) {

System.out.print(array[i][j] + " ");

}

System.out.println();

}

}

}

1. **Compare and contrast the use of for loops and while loops. When would you prefer one over the other?**

**Comparison:**

* + **For Loop:**
    - Use when the number of iterations is known beforehand.
    - Compact syntax includes initialization, condition, and increment.
  + **While Loop:**
    - Use when the number of iterations depends on a condition.
    - Flexible but requires explicit initialization and increment.

**Example:**

* + Use a **for loop** to traverse an array of fixed size.
  + Use a **while loop** to process input until a user enters a specific value.

1. **Analyze the impact of using a break statement in nested loops. What considerations should be made?** 
   * The break statement exits the innermost loop immediately, skipping remaining iterations.
   * In nested loops, it only breaks the loop where it is applied. If breaking out of multiple loops is required, consider using labeled loops or restructuring the code.

**Example:**

java

Copy code public class Main { public static void main(String[] args) { outer:

for (int i = 0; i < 3; i++) { for (int j = 0; j < 3; j++) {

if (j == 1) break outer; // Breaks the outer loop

System.out.println("i = " + i + ", j = " + j);

}

}

}

}

**11. How do arrays improve the organization and management of data in Java? Discuss with examples.**

• **Advantages:**

* Efficient data storage in contiguous memory locations. o Simplifies data access using indices.
* Allows batch processing with loops.

**Example:**

java

Copy code public class Main { public static void main(String[] args) { int[] marks = {85, 90, 78, 92}; for (int mark : marks) {

System.out.println(mark);

}

}

}

1. **Examine the differences in memory allocation for single-dimensional and multi-dimensional arrays.** 
   * **Single-dimensional array:** Allocates memory in a contiguous block for all elements.
   * **Multi-dimensional array:** Allocates memory as an array of arrays, which may not be contiguous.

**Example:**

java

Copy code

int[] singleArray = {1, 2, 3}; // Single block int[][] multiArray = { {1, 2}, {3, 4} }; // Separate blocks for each row

1. **Discuss the potential pitfalls of using uninitialized arrays in Java.** 
   * Uninitialized arrays lead to NullPointerException if accessed.
   * Must explicitly allocate memory using new or directly initialize.

**Example:**

java

Copy code

int[] array; // Declaration without initialization

// System.out.println(array[0]); // Throws NullPointerException

1. **Create a method that accepts an array and returns the maximum value using a for loop.** java

Copy code public class Main { public static int findMax(int[] array) { int max = array[0]; for (int num : array) { if (num > max) { max = num;

}

}

return max;

}

public static void main(String[] args) { int[] numbers = {5, 12, 7, 25, 9};

System.out.println("Maximum value: " + findMax(numbers));

}

}

**15. Write a Java program that finds the average of numbers stored in an integer array.** java

Copy code public class Main { public static void main(String[] args) { int[] numbers = {10, 20, 30, 40, 50}; int sum = 0;

for (int num : numbers) { sum += num;

}

double average = (double) sum / numbers.length;

System.out.println("Average: " + average);

}

}

**16. Write a Java program that sums the elements of a 2D array.**

java

Copy code public class Main { public static void main(String[] args) {

int[][] array = { {1, 2, 3},

{4, 5, 6},

{7, 8, 9}

};

int sum = 0;

for (int[] row : array) { for (int element : row) { sum += element;

}

}

System.out.println("Sum of 2D array elements: " + sum);

}

}

**17. Demonstrate how to find the minimum and maximum values in a given array.**

java

Copy code public class Main { public static void main(String[] args) { int[] numbers = {5, 3, 9, 1, 7}; int min = numbers[0]; int max = numbers[0];

for (int num : numbers) { if (num < min) { min = num;

}

if (num > max) { max = num;

}

}

System.out.println("Minimum value: " + min);

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

}

}

1. **Discuss the benefits and drawbacks of using static arrays versus dynamic arrays in Java.**

**Static Arrays:**

* + **Benefits:** Fixed size, faster memory allocation, easier to implement.
  + **Drawbacks:** Limited flexibility; size must be known beforehand.

**Dynamic Arrays (e.g., ArrayList):**

* + **Benefits:** Resizable, better for scenarios where data size is variable.
  + **Drawbacks:** Slightly slower due to resizing and dynamic memory allocation.

1. **Design a Java program that merges two sorted arrays into a single sorted array.**

java

Copy code

import java.util.Arrays;

public class Main { public static int[] mergeSortedArrays(int[] arr1, int[] arr2) { int[] merged = new int[arr1.length + arr2.length]; int i = 0, j = 0, k = 0;

while (i < arr1.length && j < arr2.length) { if (arr1[i] < arr2[j]) { merged[k++] = arr1[i++];

} else {

merged[k++] = arr2[j++];

}

}

while (i < arr1.length) { merged[k++] = arr1[i++];

}

while (j < arr2.length) { merged[k++] = arr2[j++];

}

return merged;

}

public static void main(String[] args) { int[] array1 = {1, 3, 5}; int[] array2 = {2, 4, 6}; int[] result = mergeSortedArrays(array1, array2);

System.out.println("Merged array: " + Arrays.toString(result));

}

}

**20. Write a Java program to reverse the array.**

java

Copy code

import java.util.Arrays;

public class Main { public static void main(String[] args) { int[] array = {1, 2, 3, 4, 5}; int left = 0, right = array.length - 1;

while (left < right) { int temp = array[left]; array[left] = array[right]; array[right] = temp;

left++; right--;

}

System.out.println("Reversed array: " + Arrays.toString(array));

}

}

**21. Find the second largest element in Java.**

java

Copy code

import java.util.Arrays;

public class Main { public static void main(String[] args) { int[] array = {10, 20, 15, 25, 5};

Arrays.sort(array);

int secondLargest = array[array.length - 2];

System.out.println("Second largest element: " + secondLargest);

}

}

**22. Find the first even number in a list and break the loop when it is found.**

java

Copy code public class Main { public static void main(String[] args) { int[] numbers = {1, 3, 5, 7, 8, 10};

for (int num : numbers) { if (num % 2 == 0) {

System.out.println("First even number: " + num); break;

}

}

}

}

**23. Print all odd numbers from 1 to 20 using continue to skip even numbers.**

java

Copy code public class Main { public static void main(String[] args) { for (int i = 1; i <= 20; i++) { if (i % 2 == 0) { continue;

}

System.out.println(i);

}

}

}

**24. Prompt the user to enter numbers until they enter a negative number.**

java

Copy code import java.util.Scanner;

public class Main {

public static void main(String[] args) { Scanner scanner = new Scanner(System.in); int number;

while (true) {

System.out.print("Enter a number: "); number = scanner.nextInt(); if (number < 0) {

System.out.println("You entered a negative number. Exiting."); break;

}

}

}

}

**25. Print a multiplication table but skip the multiplication by 5.** java

Copy code public class Main { public static void main(String[] args) { int number = 7; // Example table for 7 for (int i = 1; i <= 10; i++) { if (i == 5) {

continue; // Skip multiplication by 5

}

System.out.println(number + " x " + i + " = " + (number \* i));

}

}

}

**26. Program counts from 1 to 10 but breaks when it reaches 6.**

java

Copy code public class Main { public static void main(String[] args) { for (int i = 1; i <= 10; i++) { if (i == 6) { break;

}

System.out.println(i);

}

}

}

**27. Program prints numbers from 1 to 10 but skips the number 5.**

java

Copy code public class Main { public static void main(String[] args) { for (int i = 1; i <= 10; i++) { if (i == 5) { continue;

}

System.out.println(i);

}

}

}

**28. Check if a given number is prime.**

java

Copy code import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in); System.out.print("Enter a number: "); int number = scanner.nextInt(); boolean isPrime = true;

if (number <= 1) { isPrime = false;

} else { for (int i = 2; i <= number / 2; i++) { if (number % i == 0) { isPrime = false; break;

}

}

}

System.out.println(number + " is " + (isPrime ? "prime" : "not prime"));

}

}

**29. Reverse the digits of a given integer using a while loop.** java

Copy code import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in); System.out.print("Enter a number: "); int number = scanner.nextInt(); int reversed = 0;

while (number != 0) { int digit = number % 10; reversed = reversed \* 10 + digit; number /= 10;

}

System.out.println("Reversed number: " + reversed);

}

}

**30. Print the multiplication table for a given number and range.**

java

Copy code import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in); System.out.print("Enter the number: "); int number = scanner.nextInt(); System.out.print("Enter the range: "); int range = scanner.nextInt();

for (int i = 1; i <= range; i++) {

System.out.println(number + " x " + i + " = " + (number \* i));

}

}

}

**31. Count the number of vowels and consonants in a given string.**

java

Copy code import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter a string: ");

String input = scanner.nextLine().toLowerCase(); int vowels = 0, consonants = 0;

for (char c : input.toCharArray()) { if (c >= 'a' && c <= 'z') { if ("aeiou".indexOf(c) != -1) { vowels++;

} else {

consonants++;

}

}

}

System.out.println("Vowels: " + vowels);

System.out.println("Consonants: " + consonants);

}

}

**32. Print the pattern:**

Copy code

1 1 1 1 1

1 1 1 1

1 1 1

1 1

1 java Copy code public class Main { public static void main(String[] args) {

for (int i = 5; i >= 1; i--) { for (int j = 1; j <= i; j++) {

System.out.print("1 ");

}

System.out.println();

}

}

}

**33. Feedback collection system for ratings from 1 to 5.**

java

Copy code import java.util.Scanner;

public class Main {

public static void main(String[] args) { Scanner scanner = new Scanner(System.in); int totalRatings = 0, count = 0, rating;

while (true) {

System.out.print("Enter rating (1 to 5, or 0 to exit): "); rating = scanner.nextInt(); if (rating == 0) { break;

}

if (rating >= 1 && rating <= 5) { totalRatings += rating; count++;

} else {

System.out.println("Invalid rating. Please enter a number between 1 and 5.");

}

}

if (count > 0) {

System.out.println("Average rating: " + (double) totalRatings / count);

System.out.println("Total ratings received: " + count);

} else {

System.out.println("No ratings received.");

}

}

}

**34. Track user's monthly expenses.**

java

Copy code import java.util.Scanner;

public class Main {

public static void main(String[] args) { Scanner scanner = new Scanner(System.in); double totalExpenses = 0;

while (true) {

System.out.print("Enter an expense category or type 'done': "); String input = scanner.next(); if (input.equalsIgnoreCase("done")) { break;

}

System.out.print("Enter the expense amount for " + input + ": "); double amount = scanner.nextDouble(); totalExpenses += amount;

}

System.out.println("Total expenses for the month: " + totalExpenses);

}

}

**35. Password validation system.**

java

Copy code import java.util.Scanner;

public class Main {

public static void main(String[] args) { Scanner scanner = new Scanner(System.in); while (true) {

System.out.print("Create a password: ");

String password = scanner.nextLine();

if (password.length() >= 8 && password.matches(".\*[A-Z].\*") && password.matches(".\*[a-z].\*") && password.matches(".\*[0-9].\*") && password.matches(".\*[!@#$%^&\*()].\*")) {

System.out.println("Password successfully created!"); break;

} else {

System.out.println("Password must be at least 8 characters long, include an uppercase letter, "

+ "a lowercase letter, a number, and a special character.");

}

}

}

}

**36. Fitness app for daily steps logging.** java

Copy code import java.util.Scanner;

public class Main {

public static void main(String[] args) { Scanner scanner = new Scanner(System.in); int[] steps = new int[7]; int totalSteps = 0;

for (int i = 0; i < 7; i++) {

System.out.print("Enter steps for day " + (i + 1) + ": "); steps[i] = scanner.nextInt(); totalSteps += steps[i];

}

System.out.println("Total steps: " + totalSteps);

System.out.println("Average steps per day: " + (totalSteps / 7));

}

}

**37. Temperature conversion tool.**

java

Copy code import java.util.Scanner;

public class Main {

public static void main(String[] args) { Scanner scanner = new Scanner(System.in); while (true) {

System.out.print("Enter temperature (e.g., 36C or 98F, or 'exit' to quit): "); String input = scanner.nextLine(); if (input.equalsIgnoreCase("exit")) { break;

}

try {

double value = Double.parseDouble(input.substring(0, input.length() - 1)); char scale = input.charAt(input.length() - 1);

if (scale == 'C' || scale == 'c') { double fahrenheit = (value \* 9/5) + 32;

System.out.println("Converted to Fahrenheit: " + fahrenheit + "F");

} else if (scale == 'F' || scale == 'f') { double celsius = (value - 32) \* 5/9;

System.out.println("Converted to Celsius: " + celsius + "C");

} else {

System.out.println("Invalid input format.");

}

} catch (Exception e) {

System.out.println("Invalid input format.");

}

}

}

}

**38. Banking system with deposit and withdrawal.**

java

Copy code import java.util.Scanner;

public class Main {

public static void main(String[] args) { Scanner scanner = new Scanner(System.in); double balance = 0;

String transactionHistory = "";

while (true) {

System.out.println("1. Deposit");

System.out.println("2. Withdraw");

System.out.println("3. Exit"); System.out.print("Choose an option: "); int choice = scanner.nextInt();

if (choice == 1) {

System.out.print("Enter deposit amount: "); double amount = scanner.nextDouble(); balance += amount; transactionHistory += "Deposited: " + amount + "\n";

} else if (choice == 2) {

System.out.print("Enter withdrawal amount: "); double amount = scanner.nextDouble(); if (amount <= balance) {

balance -= amount; transactionHistory += "Withdrew: " + amount + "\n";

} else {

System.out.println("Insufficient balance!");

}

} else if (choice == 3) { break;

} else {

System.out.println("Invalid option!");

}

}

System.out.println("Final balance: " + balance);

System.out.println("Transaction history:");

System.out.println(transactionHistory);

}

}

**39. Grade input system for students.**

java

Copy code import java.util.Scanner;

public class Main {

public static void main(String[] args) { Scanner scanner = new Scanner(System.in); int totalGrades = 0, count = 0, highestGrade = Integer.MIN\_VALUE, passingCount = 0;

while (true) {

System.out.print("Enter grade (-1 to stop): "); int grade = scanner.nextInt(); if (grade == -1) { break;

}

totalGrades += grade; count++; if (grade > highestGrade) { highestGrade = grade;

}

if (grade >= 40) { // Assuming 40 is the passing grade passingCount++;

}

}

if (count > 0) {

System.out.println("Average grade: " + (totalGrades / count));

System.out.println("Highest grade: " + highestGrade);

System.out.println("Students who passed: " + passingCount);

} else {

System.out.println("No grades entered.");

}

}

}

**40. Shopping cart application.**

java

Copy code import java.util.Scanner;

public class Main {

public static void main(String[] args) { Scanner scanner = new Scanner(System.in); double totalAmount = 0;

String cart = "";

while (true) {

System.out.print("Enter item name (or 'checkout' to finish): "); String item = scanner.nextLine(); if (item.equalsIgnoreCase("checkout")) { break;

}

System.out.print("Enter price for " + item + ": "); double price = scanner.nextDouble(); scanner.nextLine(); // Consume newline

totalAmount += price; cart += item + " ($" + price + ")\n";

}

System.out.println("Items purchased:"); System.out.println(cart);

System.out.println("Total amount: $" + totalAmount);

}

}

**41. Calculate total sales and commission.**

java

Copy code import java.util.Scanner;

public class Main {

public static void main(String[] args) { Scanner scanner = new Scanner(System.in); double totalSales = 0; int count = 0;

while (true) {

System.out.print("Enter sales amount (negative to stop): "); double sales = scanner.nextDouble();

if (sales < 0) { break;

}

totalSales += sales; count++;

}

if (count > 0) {

System.out.println("Total sales: $" + totalSales);

System.out.println("Average sales per person: $" + (totalSales / count));

} else {

System.out.println("No sales data entered.");

}

}

}

1. **Reverse a string.** java

Copy code public class Main {

public static void main(String[] args) {

String input = "Hello World";

String reversed = new StringBuilder(input).reverse().toString();

System.out.println("Reversed string: " + reversed);

}

}

1. **Check if a string is a palindrome.** java

Copy code public class Main {

public static void main(String[] args) {

String input = "racecar";

String reversed = new StringBuilder(input).reverse().toString();

System.out.println("Is palindrome: " + input.equals(reversed));

}

}

1. **Count occurrences of each character in a string.**

java

Copy code import java.util.HashMap;

public class Main {

public static void main(String[] args) {

String input = "hello world";

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

for (char c : input.toCharArray()) { charCount.put(c, charCount.getOrDefault(c, 0) + 1);

}

System.out.println("Character occurrences: " + charCount);

}

}

**45. Reverse a given string without using the built-in reverse method.**

java

Copy code public class Main {

public static void main(String[] args) {

String input = "Hello";

String reversed = "";

for (int i = input.length() - 1; i >= 0; i--) { reversed += input.charAt(i);

}

System.out.println("Reversed string: " + reversed);

}

}

**46. Check if a given string is a palindrome.**

java

Copy code public class Main { public static boolean isPalindrome(String str) { int left = 0, right = str.length() - 1; while (left < right) { if (str.charAt(left) != str.charAt(right)) { return false;

}

left++; right--;

}

return true;

}

public static void main(String[] args) {

String input = "racecar";

System.out.println("Is palindrome: " + isPalindrome(input));

}

}

**47. Count the number of vowels and consonants in a string.**

java

Copy code public class Main {

public static void main(String[] args) { String input = "Hello World"; int vowels = 0, consonants = 0;

for (char c : input.toLowerCase().toCharArray()) { if ("aeiou".indexOf(c) != -1) { vowels++;

} else if (c >= 'a' && c <= 'z') { consonants++;

}

}

System.out.println("Vowels: " + vowels);

System.out.println("Consonants: " + consonants);

}

}

**48. Capitalize the first letter of each word in a string.** java

Copy code public class Main {

public static void main(String[] args) { String input = "hello world";

String[] words = input.split(" ");

StringBuilder capitalized = new StringBuilder();

for (String word : words) { capitalized.append(Character.toUpperCase(word.charAt(0)))

.append(word.substring(1))

.append(" ");

}

System.out.println(capitalized.toString().trim());

}

}

**49. Check if two strings are anagrams of each other.**

java

Copy code

import java.util.Arrays;

public class Main { public static boolean areAnagrams(String str1, String str2) { char[] arr1 = str1.toCharArray(); char[] arr2 = str2.toCharArray();

Arrays.sort(arr1);

Arrays.sort(arr2);

return Arrays.equals(arr1, arr2);

}

public static void main(String[] args) {

String str1 = "listen";

String str2 = "silent";

System.out.println("Are anagrams: " + areAnagrams(str1, str2));

}

}

**50. Remove duplicate characters from a string while maintaining order.**

java

Copy code import java.util.LinkedHashSet;

public class Main {

public static void main(String[] args) {

String input = "programming";

LinkedHashSet<Character> set = new LinkedHashSet<>();

for (char c : input.toCharArray()) { set.add(c);

}

StringBuilder result = new StringBuilder(); for (char c : set) { result.append(c);

}

System.out.println("String without duplicates: " + result);

}

}

**51. Find the first non-repeating character in a string.** java

Copy code import java.util.LinkedHashMap; import java.util.Map;

public class Main {

public static void main(String[] args) {

String input = "swiss";

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

for (char c : input.toCharArray()) { map.put(c, map.getOrDefault(c, 0) + 1);

}

for (Map.Entry<Character, Integer> entry : map.entrySet()) { if (entry.getValue() == 1) {

System.out.println("First non-repeating character: " + entry.getKey()); return;

}

}

System.out.println("No non-repeating characters found.");

}

}

**52. Compress a string using counts of repeated characters.**

java

Copy code public class Main {

public static String compressString(String str) { StringBuilder compressed = new StringBuilder(); int count = 1;

for (int i = 1; i < str.length(); i++) { if (str.charAt(i) == str.charAt(i - 1)) { count++;

} else {

compressed.append(str.charAt(i - 1)).append(count); count = 1;

}

}

compressed.append(str.charAt(str.length() - 1)).append(count);

return compressed.length() < str.length() ? compressed.toString() : str;

}

public static void main(String[] args) {

String input = "aabcccccaaa";

System.out.println("Compressed string: " + compressString(input));

}

}

1. **Append "World" to a StringBuffer containing "Hello".**

java

Copy code public class Main {

public static void main(String[] args) { StringBuffer sb = new StringBuffer("Hello"); sb.append(" World");

System.out.println(sb.toString());

}

}

1. **Insert "Beautiful " at index 6 in the StringBuffer containing "Hello World".**

java

Copy code public class Main {

public static void main(String[] args) {

StringBuffer sb = new StringBuffer("Hello World"); sb.insert(6, "Beautiful ");

System.out.println(sb.toString());

}

}

1. **Reverse a StringBuffer initialized with "Java Programming".**

java

Copy code public class Main {

public static void main(String[] args) {

StringBuffer sb = new StringBuffer("Java Programming"); sb.reverse();

System.out.println(sb.toString());

}

}

1. **Remove "World" from "Hello World".**

java

Copy code public class Main {

public static void main(String[] args) { StringBuffer sb = new StringBuffer("Hello World"); sb.delete(6, 11);

System.out.println(sb.toString());

}

}

1. **Reverse a StringBuffer initialized with "Java Programming".**

java

Copy code public class Main {

public static void main(String[] args) {

StringBuffer sb = new StringBuffer("Java Programming"); sb.reverse();

System.out.println(sb.toString());

}

}

1. **Delete "World" from a StringBuffer.**

java

Copy code public class Main {

public static void main(String[] args) { StringBuffer sb = new StringBuffer("Hello World"); sb.delete(6, 11);

System.out.println(sb.toString());

}

}

1. **Replace "Java" with "Python" in "I love Java programming".**

java

Copy code public class Main {

public static void main(String[] args) {

StringBuffer sb = new StringBuffer("I love Java programming"); int start = sb.indexOf("Java"); sb.replace(start, start + 4, "Python");

System.out.println(sb.toString());

}

}

1. **Create a StringBuffer, check initial capacity, and exceed it.**

java

Copy code public class Main {

public static void main(String[] args) {

StringBuffer sb = new StringBuffer(10); // Initial capacity

System.out.println("Initial capacity: " + sb.capacity());

sb.append("12345678901"); // Exceed capacity

System.out.println("New capacity: " + sb.capacity());

}

}

1. **Convert a StringBuffer to a String.** java

Copy code public class Main {

public static void main(String[] args) {

StringBuffer sb = new StringBuffer("Hello World");

String str = sb.toString();

System.out.println(str);

}

}

1. **Count vowels in a StringBuffer.**

java

Copy code public class Main {

public static void main(String[] args) { StringBuffer sb = new StringBuffer("Hello World"); int count = 0;

for (int i = 0; i < sb.length(); i++) { char c = Character.toLowerCase(sb.charAt(i)); if ("aeiou".indexOf(c) != -1) { count++;

}

}

System.out.println("Number of vowels: " + count);

}

}

1. **Trim extra spaces from both ends of a StringBuffer.**

java

Copy code public class Main {

public static void main(String[] args) {

StringBuffer sb = new StringBuffer(" Hello World ");

String trimmed = sb.toString().trim();

System.out.println(trimmed);

}

}

1. **Merge two StringBuffer objects with a space in between.**

java

Copy code public class Main {

public static void main(String[] args) {

StringBuffer sb1 = new StringBuffer("Hello");

StringBuffer sb2 = new StringBuffer("World");

StringBuffer merged = new StringBuffer(sb1).append(" ").append(sb2);

System.out.println(merged.toString());

}

}