|  |  |
| --- | --- |
|  | ***Patuakhali Science and Technology University*** |

Assignment on

***“******Java Basic Exercises[150 Exercises]”***

Course Code: CCE-122

Course Title: Object Oriented Programming Sessional

Level - I; Semester - II

|  |
| --- |
| **Submitted By**  **Name: *Md. Arif***  **ID:** 2202032; **REG:** 11223  **Session:** 2022-2023(CSE-20)  Faculty of Computer Science and Engineering |

|  |
| --- |
| **Submitted To**  Prof. Dr. Md. Samsuzzaman  Professor of Computer and Communication Engineering Department  Faculty of Computer Science and Engineering |

Submission Date: 17 July 2024

Lab Problem No: 02

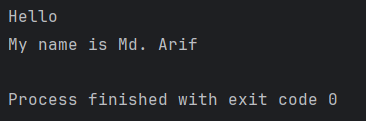
**Java Basic Exercises[150 Exercises]**

**1.** Write a Java program to print 'Hello' on screen and your name on a separate line.  
*Expected Output* :  
Hello  
My name is Md. Arif

Code:

|  |
| --- |
| package W3BasicJava150;  public class \_1\_hello {  public static void main(String[] args) {  System.out.println("Hello");  System.out.println("My name is Md. Arif");  }  } |

Output:



**2.** Write a Java program to print the sum of two numbers.  
Test Data:  
74 + 36  
*Expected Output* :  
110

Code:

|  |
| --- |
| package W3BasicJava150;  public class \_2\_Sum {  public static void main(String[] args) {  System.out.println(74+36);  }  } |

Output:

A black background with white text

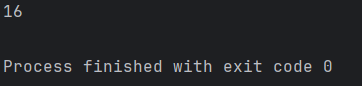
Description automatically generated

**3.** Write a Java program to divide two numbers and print them on the screen.  
Test Data :  
50/3  
*Expected Output* :  
16

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| package W3BasicJava150;  public class \_3\_Div {  public static void main(String[] args) {  int div=50/3;  System.out.println(div);  }  } |

Output:



**4.** Write a Java program to print the results of the following operations.  
*Test Data:*  
a. -5 + 8 \* 6  
b. (55+9) % 9  
c. 20 + -3\*5 / 8  
d. 5 + 15 / 3 \* 2 - 8 % 3  
*Expected Output* :  
43  
1  
19  
13

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| package W3BasicJava150;  public class \_4\_operation {  public static void main(String[] args) {  System.out.println(-5 + 8 \* 6);  System.out.println((55+9) % 9);  System.out.println(20 + -3\*5 / 8);  System.out.println(5 + 15 / 3 \* 2 - 8 % 3);  }  } |

Output:

A black screen with white text

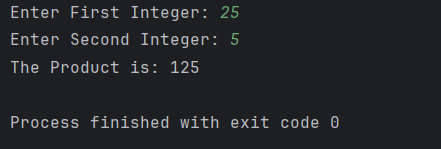
Description automatically generated

**5.** Write a Java program that takes two numbers as input and displays the product of two numbers.  
*Test Data:*  
Input first number: 25  
Input second number: 5  
*Expected Output* :  
25 x 5 = 125

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| package W3BasicJava150;  import java.util.Scanner;  public class \_5\_Product {  public static void main(String[] args) {  Scanner in = new Scanner(System.in);  System.out.print("Enter First Integer: ");  int i1=in.nextInt();  System.out.print("Enter Second Integer: ");  int i2=in.nextInt();  int p=i1\*i2;  System.out.println("The Product is: "+p);  }  } |

Output:

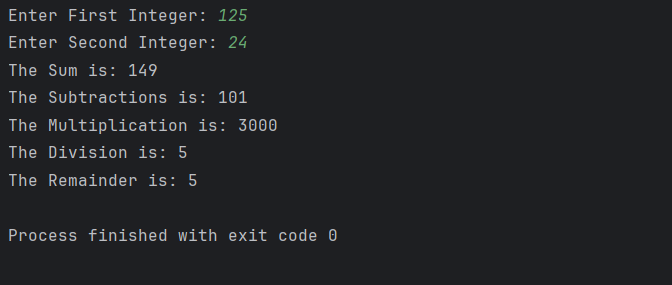


**6.** Write a Java program to print the sum (addition), multiply, subtract, divide and remainder of two numbers.  
*Test Data:*  
Input first number: 125  
Input second number: 24  
*Expected Output* :  
125 + 24 = 149  
125 - 24 = 101  
125 x 24 = 3000  
125 / 24 = 5  
125 mod 24 = 5

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| package W3BasicJava150;  import java.util.Scanner;  public class \_6\_ASMDR{  public static void main(String[] args) {  Scanner in = new Scanner(System.in);  System.out.print("Enter First Integer: ");  int i1=in.nextInt();  System.out.print("Enter Second Integer: ");  int i2=in.nextInt();  int sum=i1+i2;  System.out.println("The Sum is: "+sum);  int sub=i1-i2;  System.out.println("The Subtractions is: "+sub);  int m=i1\*i2;  System.out.println("The Multiplication is: "+m);  int d=i1/i2;  System.out.println("The Division is: "+d);  int r=i1%i2;  System.out.println("The Remainder is: "+r);  }  } |

Output:



**7.** Write a Java program that takes a number as input and prints its multiplication table up to 10.  
*Test Data:*  
Input a number: 8  
*Expected Output* :  
8 x 1 = 8  
8 x 2 = 16  
8 x 3 = 24  
...  
8 x 10 = 80

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| package W3BasicJava150;  import java.util.Scanner;  public class \_7\_Multiplication {  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  System.out.print("Enter an integer number: ");  int num = sc.nextInt();  System.out.println("Multiplication Table of "+num);  for (int i=1; i<=10; i++ )  {  System.out.println(num+" \* "+i+" = "+num\*i);  }  }  } |

Output:

A screenshot of a computer

Description automatically generated

**8.** Write a Java program to display the following pattern.  
*Sample Pattern :*

J a v v a

J a a v v a a

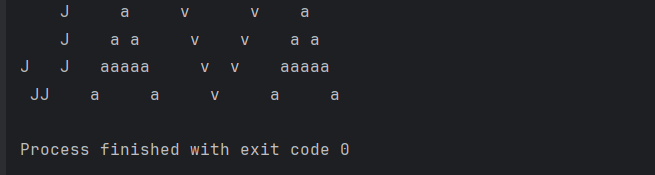
J J aaaaa V V aaaaa

JJ a a V a a

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| package W3BasicJava150;  public class \_8\_javapattern {  public static void main(String[] args) {  System.out.println(" J a v v a");  System.out.println(" J a a v v a a");  System.out.println("J J aaaaa v v aaaaa");  System.out.println(" JJ a a v a a");  }  } |

Output:



**9.** Write a Java program to compute the specified expressions and print the output.  
*Test Data:*  
((25.5 \* 3.5 - 3.5 \* 3.5) / (40.5 - 4.5))  
*Expected Output*  
2.138888888888889

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| package W3BasicJava150;  public class \_9\_expretion {  public static void main(String[] args) {  double x=((25.5 \* 3.5 - 3.5 \* 3.5) / (40.5 - 4.5));  System.out.println(x);  }  } |

Output:

A black background with white text

Description automatically generated

**10.** Write a Java program to compute a specified formula.  
*Specified Formula :*  
4.0 \* (1 - (1.0/3) + (1.0/5) - (1.0/7) + (1.0/9) - (1.0/11))  
*Expected Output*  
2.9760461760461765

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| package W3BasicJava150;  public class \_10\_specified\_expretion {  public static void main(String[] args) {  double x=4.0 \* (1 - (1.0/3) + (1.0/5) - (1.0/7) + (1.0/9) - (1.0/11));  System.out.println(x);  }  } |

Output:

A black background with white text

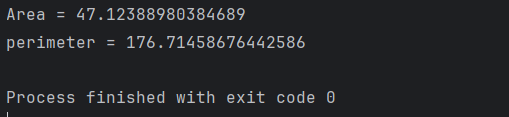
Description automatically generated

**11.** Write a Java program to print the area and perimeter of a circle.  
*Test Data:*  
Radius = 7.5  
*Expected Output*  
Perimeter is = 47.12388980384689  
Area is = 176.71458676442586

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| package W3BasicJava150;  public class \_11\_Circle {  public static void main(String[] args) {  double radius, area, perimeter;  radius = 7.5;  area = 2 \* Math.PI \* radius;  perimeter = Math.PI \* radius \* radius;  System.out.println("Area = "+area);  System.out.println("perimeter = "+perimeter);  }  } |

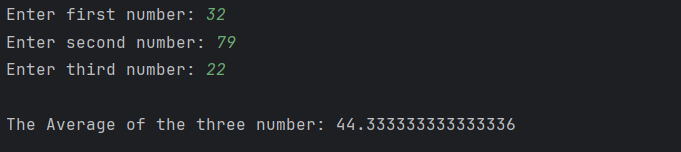
Output:



**12.** Write a Java program that takes three numbers as input to calculate and print the average of the numbers.  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| package W3BasicJava150;  import java.util.Scanner;  public class \_12\_ThreeNumber {  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  double n1, n2, n3, avg;  System.out.print("Enter first number: ");  n1 = sc.nextInt();  System.out.print("Enter second number: ");  n2 = sc.nextInt();  System.out.print("Enter third number: ");  n3 = sc.nextInt();  avg = (n1+n2+n3)/3;  System.out.println("\nThe Average of the three number: "+avg);  }  } |

Output:



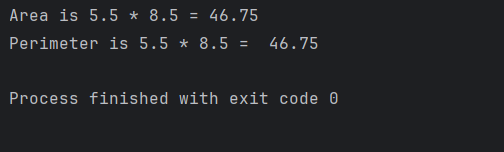
**13.** Write a Java program to print the area and perimeter of a rectangle.  
*Test Data:*  
Width = 5.5 Height = 8.5

*Expected Output*  
Area is 5.6 \* 8.5 = 47.60  
Perimeter is 2 \* (5.6 + 8.5) = 28.20

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| package W3BasicJava150;  public class \_13\_Rectangle {  public static void main(String[] args) {  double h, w, area, perimeter;  h = 5.5;  w = 8.5;  area = h \* w;  perimeter = 2\*(h + w);  System.out.println("Area is "+h+" \* "+w+" = " +area);  System.out.println("Perimeter is 5.5 \* 8.5 = "+area);  }  } |

Output:



**14.** Write a Java program to print an American flag on the screen.  
*Expected Output*

\* \* \* \* \* \* ==================================

\* \* \* \* \* ==================================

\* \* \* \* \* \* ==================================

\* \* \* \* \* ==================================

\* \* \* \* \* \* ==================================

\* \* \* \* \* ==================================

\* \* \* \* \* \* ==================================

\* \* \* \* \* ==================================

\* \* \* \* \* \* ==================================

==============================================

==============================================

==============================================

==============================================

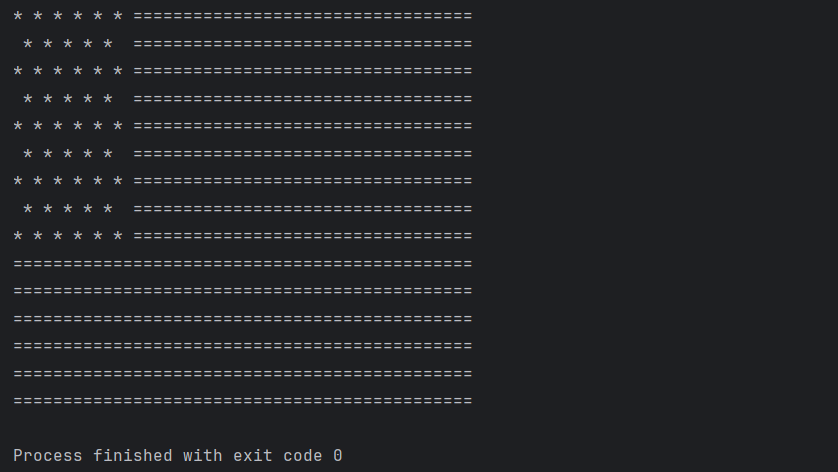
==============================================

==============================================

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| package W3BasicJava150;  public class \_14\_Flag {  public static void main(String[] args) {  System.out.println("\* \* \* \* \* \* ==================================");  System.out.println(" \* \* \* \* \* ==================================");  System.out.println("\* \* \* \* \* \* ==================================");  System.out.println(" \* \* \* \* \* ==================================");  System.out.println("\* \* \* \* \* \* ==================================");  System.out.println(" \* \* \* \* \* ==================================");  System.out.println("\* \* \* \* \* \* ==================================");  System.out.println(" \* \* \* \* \* ==================================");  System.out.println("\* \* \* \* \* \* ==================================");  System.out.println("==============================================");  System.out.println("==============================================");  System.out.println("==============================================");  System.out.println("==============================================");  System.out.println("==============================================");  System.out.println("==============================================");  }  } |

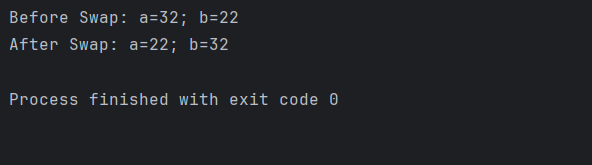
Output:



**15.** Write a Java program to swap two variables.  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| package W3BasicJava150;  public class \_15\_Swap {  public static void main(String[] args) {  int a, b, temp;  a=32;  b=22;  System.out.println("Before Swap: a="+a+"; b="+b);  temp=a;  a=b;  b=temp;  System.out.println("After Swap: a="+a+"; b="+b);  }  } |

Output:



**16.** Write a Java program to print a face.  
*Expected Output*

+"""""+

[| o o |]

| ^ |

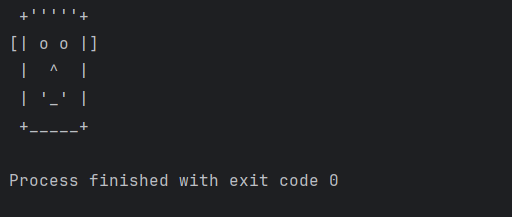
| '-' |

+-----+

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| package W3BasicJava150;  public class \_16\_Face {  public static void main(String[] args) {  System.out.println(" +'''''+ ");  System.out.println("[| o o |]");  System.out.println(" | ^ | ");  System.out.println(" | '\_' | ");  System.out.println(" +\_\_\_\_\_+ ");  }  } |

Output:



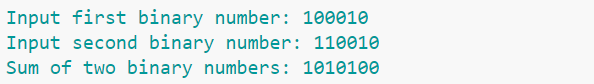
**17.** Write a Java program to add two binary numbers.  
Input Data:  
Input first binary number: 10  
Input second binary number: 11  
*Expected Output*

Sum of two binary numbers: 101

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.Scanner;  public class Exercise17 {  public static void main(String[] args) {  // Declare variables to store two binary numbers, an index, and a remainder  long binary1, binary2;  int i = 0, remainder = 0;    // Create an array to store the sum of binary digits  int[] sum = new int[20];    // Create a Scanner object to read input from the user  Scanner in = new Scanner(System.in);  // Prompt the user to input the first binary number  System.out.print("Input first binary number: ");  binary1 = in.nextLong();    // Prompt the user to input the second binary number  System.out.print("Input second binary number: ");  binary2 = in.nextLong();  // Perform binary addition while there are digits in the binary numbers  while (binary1 != 0 || binary2 != 0)  {  // Calculate the sum of binary digits and update the remainder  sum[i++] = (int)((binary1 % 10 + binary2 % 10 + remainder) % 2);  remainder = (int)((binary1 % 10 + binary2 % 10 + remainder) / 2);  binary1 = binary1 / 10;  binary2 = binary2 / 10;  }    // If there is a remaining carry, add it to the sum  if (remainder != 0) {  sum[i++] = remainder;  }    // Decrement the index to prepare for printing  --i;    // Display the sum of the two binary numbers  System.out.print("Sum of two binary numbers: ");  while (i >= 0) {  System.out.print(sum[i--]);  }    System.out.print("\n");  }  } |

Output:



**18.** Write a Java program to multiply two binary numbers.  
Input Data:  
Input the first binary number: 10  
Input the second binary number: 11  
*Expected Output*

Product of two binary numbers: 110

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.Scanner;  public class Exercise18 {  public static void main(String[] args) {  // Declare variables to store two binary numbers and the product  long binary1, binary2, multiply = 0;    // Initialize digit and factor variables for processing binary2  int digit, factor = 1;    // Create a Scanner object to read input from the user  Scanner in = new Scanner(System.in);    // Prompt the user to input the first binary number  System.out.print("Input the first binary number: ");  binary1 = in.nextLong();    // Prompt the user to input the second binary number  System.out.print("Input the second binary number: ");  binary2 = in.nextLong();    // Process binary2 to calculate the product  while (binary2 != 0) {  digit = (int)(binary2 % 10);  if (digit == 1) {  binary1 = binary1 \* factor;  multiply = binaryproduct((int) binary1, (int) multiply);  } else {  binary1 = binary1 \* factor;  }  binary2 = binary2 / 10;  factor = 10;  }    // Display the product of the two binary numbers  System.out.print("Product of two binary numbers: " + multiply + "\n");  }  // Method to calculate the product of two binary numbers  static int binaryproduct(int binary1, int binary2) {  int i = 0, remainder = 0;  int[] sum = new int[20];  int binary\_prod\_result = 0;  while (binary1 != 0 || binary2 != 0) {  sum[i++] = (binary1 % 10 + binary2 % 10 + remainder) % 2;  remainder = (binary1 % 10 + binary2 % 10 + remainder) / 2;  binary1 = binary1 / 10;  binary2 = binary2 / 10;  }  if (remainder != 0) {  sum[i++] = remainder;  }    --i;    while (i >= 0) {  binary\_prod\_result = binary\_prod\_result \* 10 + sum[i--];  }  return binary\_prod\_result;  }  } |

Output:

A close-up of a number

Description automatically generated**19.** Write a Java program to convert an integer number to a binary number.  
Input Data:  
Input a Decimal Number : 5  
*Expected Output*

Binary number is: 101

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.Scanner;  public class Exercise19 {  public static void main(String args[]) {  // Declare variables to store decimal number, quotient, and an array for binary digits  int dec\_num, quot, i = 1, j;  int bin\_num[] = new int[100];    // Create a Scanner object to read input from the user  Scanner scan = new Scanner(System.in);  // Prompt the user to input a decimal number  System.out.print("Input a Decimal Number: ");  dec\_num = scan.nextInt();  // Initialize the quotient with the decimal number  quot = dec\_num;  // Convert the decimal number to binary and store binary digits  while (quot != 0) {  bin\_num[i++] = quot % 2;  quot = quot / 2;  }  // Display the binary representation of the decimal number  System.out.print("Binary number is: ");  for (j = i - 1; j > 0; j--) {  System.out.print(bin\_num[j]);  }  System.out.print("\n");  }  } |

Output:

A number on a white background

Description automatically generated

**20.** Write a Java program to convert a decimal number to a hexadecimal number.  
Input Data:  
Input a decimal number: 15  
*Expected Output*

Hexadecimal number is : F

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.Scanner;  public class Exercise20 {  public static void main(String args[]) {  // Declare variables to store decimal number and remainder  int dec\_num, rem;    // Initialize an empty string for the hexadecimal number  String hexdec\_num = "";    // Define the hexadecimal number digits  char hex[] = {'0','1','2','3','4','5','6','7','8','9','A','B','C','D','E','F'};    // Create a Scanner object to read input from the user  Scanner in = new Scanner(System.in);  // Prompt the user to input a decimal number  System.out.print("Input a decimal number: ");  dec\_num = in.nextInt();  // Convert the decimal number to hexadecimal  while (dec\_num > 0) {  rem = dec\_num % 16;  hexdec\_num = hex[rem] + hexdec\_num;  dec\_num = dec\_num / 16;  }    // Display the hexadecimal representation of the decimal number  System.out.print("Hexadecimal number is: " + hexdec\_num + "\n");  }  } |

Output:

A close up of a number

Description automatically generated

**21.** Write a Java program to convert a decimal number to an octal number.  
Input Data:  
Input a Decimal Number: 15  
*Expected Output*

Octal number is: 17

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.Scanner;  public class Exercise21 {  public static void main(String args[]) {  // Declare variables to store decimal number, remainder, quotient, and an array for octal digits  int dec\_num, rem, quot, i = 1, j;  int oct\_num[] = new int[100];    // Create a Scanner object to read input from the user  Scanner scan = new Scanner(System.in);  // Prompt the user to input a decimal number  System.out.print("Input a Decimal Number: ");  dec\_num = scan.nextInt();  // Initialize the quotient with the decimal number  quot = dec\_num;  // Convert the decimal number to octal and store octal digits  while (quot != 0) {  oct\_num[i++] = quot % 8;  quot = quot / 8;  }  // Display the octal representation of the decimal number  System.out.print("Octal number is: ");  for (j = i - 1; j > 0; j--) {  System.out.print(oct\_num[j]);  }    System.out.print("\n");  }  } |

Output:

A close up of a number

Description automatically generated

**22.** Write a Java program to convert a binary number to a decimal number.  
Input Data:  
Input a binary number: 100  
*Expected Output*

Decimal Number: 4

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.Scanner;  public class Exercise22 {  public static void main(String[] args) {  // Create a Scanner object to read input from the user  Scanner sc = new Scanner(System.in);    // Declare variables to store binary and decimal numbers, remainder, and a multiplier  long binaryNumber, decimalNumber = 0, j = 1, remainder;    // Prompt the user to input a binary number  System.out.print("Input a binary number: ");  binaryNumber = sc.nextLong();  // Convert the binary number to decimal  while (binaryNumber != 0) {  remainder = binaryNumber % 10;  decimalNumber = decimalNumber + remainder \* j;  j = j \* 2;  binaryNumber = binaryNumber / 10;  }    // Display the decimal representation of the binary number  System.out.println("Decimal Number: " + decimalNumber);  }  } |

Output:



**23.** Write a Java program to convert a binary number to a hexadecimal number.  
Input Data:  
Input a Binary Number: 1101  
*Expected Output*

HexaDecimal value: D

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.Scanner;  public class Exercise23 {  public static void main(String[] args) {  // Declare an array to store hexadecimal digits, variables for calculation, and binary input  int[] hex = new int[1000];  int i = 1, j = 0, rem, dec = 0, bin;  // Create a Scanner object to read input from the user  Scanner in = new Scanner(System.in);  // Prompt the user to input a binary number  System.out.print("Input a Binary Number: ");  bin = in.nextInt();  // Convert the binary number to decimal  while (bin > 0) {  rem = bin % 2;  dec = dec + rem \* i;  i = i \* 2;  bin = bin / 10;  }  i = 0;  // Convert the decimal number to hexadecimal  while (dec != 0) {  hex[i] = dec % 16;  dec = dec / 16;  i++;  }  // Display the hexadecimal value  System.out.print("Hexadecimal value: ");  for (j = i - 1; j >= 0; j--) {  if (hex[j] > 9) {  System.out.print((char)(hex[j] + 55));  } else {  System.out.print(hex[j]);  }  }  System.out.print("\n");  }  } |

Output:

A close up of a number

Description automatically generated

**24.** Write a Java program to convert a binary number to an octal number.  
Input Data:  
Input a Binary Number: 111  
*Expected Output*

Octal number: 7

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  public class Exercise24 {  public static void main(String[] args) {  // Declare variables to store binary and decimal numbers, remainder, quotient, and an array for octal digits  int binnum, binnum1, rem, decnum = 0, quot, i = 1, j;  int octnum[] = new int[100];  // Create a Scanner object to read input from the user  Scanner scan = new Scanner(System.in);  // Prompt the user to input a binary number  System.out.print("Input a Binary Number : ");  binnum = scan.nextInt();  binnum1 = binnum;  // Convert the binary number to decimal  while (binnum > 0) {  rem = binnum % 10;  decnum = decnum + rem \* i;  i = i \* 2;  binnum = binnum / 10;  }  i = 1;  quot = decnum;  // Convert the decimal number to octal  while (quot > 0) {  octnum[i++] = quot % 8;  quot = quot / 8;  }  // Display the equivalent octal value of the original binary number  System.out.print("Equivalent Octal Value of " + binnum1 + " is :");  for (j = i - 1; j > 0; j--) {  System.out.print(octnum[j]);  }  System.out.print("\n");  }  } |

Output:



**25.** Write a Java program to convert a octal number to a decimal number.  
Input Data:  
Input any octal number: 10  
*Expected Output*

Equivalent decimal number: 8

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.Scanner;  public class Exercise25 {  public static void main(String[] args) {  // Create a Scanner object to read input from the user  Scanner in = new Scanner(System.in);    // Declare variables to store octal and decimal numbers, and an index counter  long octal\_num, decimal\_num = 0;  int i = 0;    // Prompt the user to input an octal number  System.out.print("Input any octal number: ");  octal\_num = in.nextLong();    // Convert the octal number to decimal  while (octal\_num != 0) {  decimal\_num = (long)(decimal\_num + (octal\_num % 10) \* Math.pow(8, i++));  octal\_num = octal\_num / 10;  }    // Display the equivalent decimal number  System.out.print("Equivalent decimal number: " + decimal\_num + "\n");  }  } |

Output:

A close-up of a number

Description automatically generated

**26.** Write a Java program to convert a octal number to a binary number.  
Input Data:  
Input any octal number: 7  
*Expected Output*

Equivalent binary number: 111

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.Scanner;  public class Exercise26 {  public static void main(String[] args) {  // Create a Scanner object to read input from the user  Scanner in = new Scanner(System.in);    // Define an array to map octal digits to their binary equivalents  int[] octal\_numvalues = {0, 1, 10, 11, 100, 101, 110, 111};    // Declare variables to store octal, temporary octal, and binary numbers, and a place value  long octal\_num, tempoctal\_num, binary\_num, place;  int rem;    // Prompt the user to input an octal number  System.out.print("Input any octal number: ");  octal\_num = in.nextLong();  tempoctal\_num = octal\_num;  binary\_num = 0;  place = 1;    // Convert the octal number to binary using the mapping array  while (tempoctal\_num != 0) {  rem = (int)(tempoctal\_num % 10);  binary\_num = octal\_numvalues[rem] \* place + binary\_num;  tempoctal\_num /= 10;  place \*= 1000;  }    // Display the equivalent binary number  System.out.print("Equivalent binary number: " + binary\_num + "\n");  }  } |

Output:



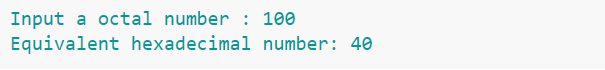
**27.** Write a Java program to convert a octal number to a hexadecimal number.  
Input Data:  
Input a octal number : 100  
*Expected Output*

Equivalent hexadecimal number: 40

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| public class Exercise27 {  public static void main(String args[]) {  // Declare variables to store octal number and its decimal and hexadecimal equivalents  String octal\_num, hex\_num;  int decnum;    // Create a Scanner object to read input from the user  Scanner in = new Scanner(System.in);    // Prompt the user to input an octal number  System.out.print("Input an octal number : ");  octal\_num = in.nextLine();    // Convert the octal number to its decimal equivalent using parseInt  decnum = Integer.parseInt(octal\_num, 8);    // Convert the decimal number to its hexadecimal equivalent  hex\_num = Integer.toHexString(decnum);    // Display the equivalent hexadecimal number  System.out.print("Equivalent hexadecimal number: " + hex\_num + "\n");  }  } |

Output:



**28.** Write a Java program to convert a hexadecimal value into a decimal number.  
Input Data:  
Input a hexadecimal number: 25  
*Expected Output*

Equivalent decimal number is: 37

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.Scanner;  public class Exercise28 {  // Function to convert a hexadecimal string to a decimal integer  public static int hex\_to\_decimal(String s) {  // Define a string containing hexadecimal digits  String digits = "0123456789ABCDEF";  s = s.toUpperCase(); // Convert the input string to uppercase  int val = 0; // Initialize the decimal value to 0  // Iterate through each character in the input string  for (int i = 0; i < s.length(); i++) {  char c = s.charAt(i); // Get the current character  int d = digits.indexOf(c); // Find the index of the character in the digits string  val = 16 \* val + d; // Update the decimal value using hexadecimal conversion  }  return val; // Return the decimal value  }  public static void main(String args[]) {  String hexdec\_num;  int dec\_num;  Scanner scan = new Scanner(System.in);  // Prompt the user to input a hexadecimal number  System.out.print("Input a hexadecimal number: ");  hexdec\_num = scan.nextLine();  // Call the hex\_to\_decimal function to convert the hexadecimal number to decimal  dec\_num = hex\_to\_decimal(hexdec\_num);  // Display the equivalent decimal number  System.out.print("Equivalent decimal number is: " + dec\_num + "\n");  }  } |

Output:

A close up of a number

Description automatically generated

**29.** Write a Java program to convert a hexadecimal number into a binary number.  
Input Data:  
Enter Hexadecimal Number : 37  
*Expected Output*

Equivalent Binary Number is: 110111

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.Scanner;  public class Exercise29 {  // Function to convert a hexadecimal string to a decimal integer  public static int hex\_to\_binary(String s) {  // Define a string containing hexadecimal digits  String digits = "0123456789ABCDEF";  s = s.toUpperCase(); // Convert the input string to uppercase  int val = 0; // Initialize the decimal value to 0  // Iterate through each character in the input string  for (int i = 0; i < s.length(); i++) {  char c = s.charAt(i); // Get the current character  int d = digits.indexOf(c); // Find the index of the character in the digits string  val = 16 \* val + d; // Update the decimal value using hexadecimal conversion  }  return val; // Return the decimal value  }  public static void main(String args[]) {  String hexdec\_num;  int dec\_num, i = 1, j;  int bin\_num[] = new int[100];  Scanner scan = new Scanner(System.in);  // Prompt the user to enter a hexadecimal number  System.out.print("Enter Hexadecimal Number : ");  hexdec\_num = scan.nextLine();  // Call the hex\_to\_binary function to convert the hexadecimal number to decimal  dec\_num = hex\_to\_binary(hexdec\_num);  // Convert the decimal number to binary  while (dec\_num != 0) {  bin\_num[i++] = dec\_num % 2;  dec\_num = dec\_num / 2;  }  // Display the equivalent binary number  System.out.print("Equivalent Binary Number is: ");  for (j = i - 1; j > 0; j--) {  System.out.print(bin\_num[j]);  }  System.out.print("\n");  }  } |

Output:

A close-up of a number

Description automatically generated

**30.** Write a Java program to convert a hexadecimal value into an octal number.  
Input Data:  
Input a hexadecimal number: 40  
*Expected Output*

Equivalent of octal number is: 100

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.Scanner;  public class Main {  // Function to convert a hexadecimal string to a decimal integer  public static int hex\_to\_oct(String s) {  // Define a string containing hexadecimal digits  String digits = "0123456789ABCDEF";  s = s.toUpperCase(); // Convert the input string to uppercase  int val = 0; // Initialize the decimal value to 0  // Iterate through each character in the input string  for (int i = 0; i < s.length(); i++) {  char c = s.charAt(i); // Get the current character  int d = digits.indexOf(c); // Find the index of the character in the digits string  val = 16 \* val + d; // Update the decimal value using hexadecimal conversion  }  return val; // Return the decimal value  }  public static void main(String args[]) {  String hexdec\_num;  int dec\_num, i = 1, j;  int octal\_num[] = new int[100];  Scanner in = new Scanner(System.in);  // Prompt the user to input a hexadecimal number  System.out.print("Input a hexadecimal number: ");  hexdec\_num = in.nextLine();  // Call the hex\_to\_oct function to convert the hexadecimal number to decimal  dec\_num = hex\_to\_oct(hexdec\_num);  // Convert the decimal number to octal  while (dec\_num != 0) {  octal\_num[i++] = dec\_num % 8;  dec\_num = dec\_num / 8;  }  // Display the equivalent octal number  System.out.print("Equivalent of octal number is: ");  for (j = i - 1; j > 0; j--) {  System.out.print(octal\_num[j]);  }  System.out.print("\n");  }  } |

Output:



**31.** Write a Java program to check whether Java is installed on your [computer](https://www.w3resource.com/java-exercises/basic/index.php).  
*Expected Output*

Java Version: 1.8.0\_71

Java Runtime Version: 1.8.0\_71-b15

Java Home: /opt/jdk/jdk1.8.0\_71/jre

Java Vendor: Oracle Corporation

Java Vendor URL: http://Java.oracle.com/

Java Class Path: .

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| public class Exercise31 {  public static void main(String[] args) {  // Display Java version  System.out.println("\nJava Version: " + System.getProperty("java.version"));    // Display Java runtime version  System.out.println("Java Runtime Version: " + System.getProperty("java.runtime.version"));    // Display Java home directory  System.out.println("Java Home: " + System.getProperty("java.home"));    // Display Java vendor name  System.out.println("Java Vendor: " + System.getProperty("java.vendor"));    // Display Java vendor URL  System.out.println("Java Vendor URL: " + System.getProperty("java.vendor.url"));    // Display Java class path  System.out.println("Java Class Path: " + System.getProperty("java.class.path") + "\n");  }  } |

Output:

A computer code with numbers and letters

Description automatically generated

**32.** Write a Java program to compare two numbers.  
Input Data:  
Input first integer: 25  
Input second integer: 39  
*Expected Output*

25 != 39

25 < 39

25 <= 39

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.Scanner;  public class Exercise32 {  public static void main(String args[]) {  // Create a Scanner to obtain input from the command window  Scanner input = new Scanner(System.in);  int number1; // First number to compare  int number2; // Second number to compare  // Prompt the user to input the first integer  System.out.print("Input first integer: ");  number1 = input.nextInt(); // Read the first number from the user  // Prompt the user to input the second integer  System.out.print("Input second integer: ");  number2 = input.nextInt(); // Read the second number from the user  // Compare and display the results  if (number1 == number2)  System.out.printf("%d == %d\n", number1, number2);  if (number1 != number2)  System.out.printf("%d != %d\n", number1, number2);  if (number1 < number2)  System.out.printf("%d < %d\n", number1, number2);  if (number1 > number2)  System.out.printf("%d > %d\n", number1, number2);  if (number1 <= number2)  System.out.printf("%d <= %d\n", number1, number2);  if (number1 >= number2)  System.out.printf("%d >= %d\n", number1, number2);  }  } |

Output:

A screenshot of a computer

Description automatically generated

**33.** Write a Java program and compute the sum of an integer's digits.  
Input Data:  
Input an integer: 25  
*Expected Output*

The sum of the digits is: 7

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.Scanner;  public class Exercise33 {  public static void main(String[] args) {  Scanner input = new Scanner(System.in);    // Prompt the user to input an integer  System.out.print("Input an integer: ");    // Read the integer from the user  long n = input.nextLong();    // Calculate and display the sum of the digits  System.out.println("The sum of the digits is: " + sumDigits(n));  }  public static int sumDigits(long n) {  int sum = 0;    // Calculate the sum of the digits  while (n != 0) {  sum += n % 10;  n /= 10;  }    return sum;  }  } |

Output:



**34.** Write a Java program to compute hexagon area.  
Area of a hexagon = (6 \* s^2)/(4\*tan(π/6))  
where s is the length of a side  
Input Data:  
Input the length of a side of the hexagon: 6  
*Expected Output*

The area of the hexagon is: 93.53074360871938

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.Scanner;  public class Exercise34 {  public static void main(String[] args) {  Scanner input = new Scanner(System.in);    // Prompt the user to input the length of a side of the hexagon  System.out.print("Input the length of a side of the hexagon: ");    // Read the length of a side from the user  double s = input.nextDouble();    // Calculate and display the area of the hexagon  System.out.print("The area of the hexagon is: " + hexagonArea(s) + "\n");  }  public static double hexagonArea(double s) {  // Calculate the area of a hexagon based on its side length  return (6 \* (s \* s)) / (4 \* Math.tan(Math.PI / 6));  }  } |

Output:



**35.** Write a Java program to compute the area of a polygon.  
Area of a polygon = (n\*s^2)/(4\*tan(π/n))  
where n is n-sided polygon and s is the length of a side  
Input Data:  
Input the number of sides on the polygon: 7  
Input the length of one of the sides: 6  
*Expected Output*

The area is: 130.82084798405722

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.Scanner;  public class Exercise35 {  public static void main(String[] args) {  Scanner input = new Scanner(System.in);    // Prompt the user to input the number of sides on the polygon  System.out.print("Input the number of sides on the polygon: ");    // Read the number of sides from the user  int ns = input.nextInt();    // Prompt the user to input the length of one of the sides  System.out.print("Input the length of one of the sides: ");    // Read the length of one side from the user  double side = input.nextDouble();    // Calculate and display the area of the polygon  System.out.print("The area is: " + polygonArea(ns, side) + "\n");  }    public static double polygonArea(int ns, double side) {  // Calculate the area of a polygon based on the number of sides and side length  return (ns \* (side \* side)) / (4.0 \* Math.tan((Math.PI / ns)));  }  } |

Output:

A close-up of a number

Description automatically generated

**36.** Write a Java program to compute the distance between two points on the earth's surface.  
Distance between the two points [ (x1,y1) & (x2,y2)]  
d = radius \* arccos(sin(x1) \* sin(x2) + cos(x1) \* cos(x2) \* cos(y1 - y2))  
Radius of the earth r = 6371.01 Kilometers  
Input Data:  
Input the latitude of coordinate 1: 25  
Input the longitude of coordinate 1: 35  
Input the latitude of coordinate 2: 35.5  
Input the longitude of coordinate 2: 25.5  
*Expected Output*

The distance between those points is: 1480.0848451069087 km

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.Scanner;  public class Exercise36 {    public static void main(String[] args) {  Scanner input = new Scanner(System.in);    // Prompt the user to input the latitude and longitude of coordinate 1  System.out.print("Input the latitude of coordinate 1: ");  double lat1 = input.nextDouble();  System.out.print("Input the longitude of coordinate 1: ");  double lon1 = input.nextDouble();    // Prompt the user to input the latitude and longitude of coordinate 2  System.out.print("Input the latitude of coordinate 2: ");  double lat2 = input.nextDouble();  System.out.print("Input the longitude of coordinate 2: ");  double lon2 = input.nextDouble();  // Calculate and display the distance between the two coordinates  System.out.print("The distance between those points is: " + distance\_Between\_LatLong(lat1, lon1, lat2, lon2) + " km\n");  }  // Points will be converted to radians before calculation  public static double distance\_Between\_LatLong(double lat1, double lon1, double lat2, double lon2) {  // Convert latitude and longitude to radians  lat1 = Math.toRadians(lat1);  lon1 = Math.toRadians(lon1);  lat2 = Math.toRadians(lat2);  lon2 = Math.toRadians(lon2);  // Earth's mean radius in kilometers  double earthRadius = 6371.01;  // Calculate the distance using the haversine formula  return earthRadius \* Math.acos(Math.sin(lat1) \* Math.sin(lat2) + Math.cos(lat1) \* Math.cos(lat2) \* Math.cos(lon1 - lon2));  }  } |

Output:

A white background with blue text

Description automatically generated

**37.** Write a Java program to reverse a string.  
Input Data:  
Input a string: The quick brown fox  
*Expected Output*

Reverse string: xof nworb kciuq ehT

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.Scanner;  public class Exercise37 {  public static void main(String[] args) {  // Create a scanner to obtain input from the user  Scanner scanner = new Scanner(System.in);    // Prompt the user to input a string  System.out.print("Input a string: ");    // Read the input string and convert it to a character array  char[] letters = scanner.nextLine().toCharArray();    // Display a message before printing the reversed string  System.out.print("Reverse string: ");    // Iterate through the character array in reverse order and print each character  for (int i = letters.length - 1; i >= 0; i--) {  System.out.print(letters[i]);  }    // Print a newline character to end the output line  System.out.print("\n");  }  } |

Output:

A close-up of words

Description automatically generated**38.** Write a Java program to count letters, spaces, numbers and other characters in an input string.  
*Expected Output*

The string is : Aa kiu, I swd skieo 236587. GH kiu: sieo?? 25.33

letter: 23

space: 9

number: 10

other: 6

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.Scanner;  public class Exercise38 {  public static void main(String[] args) {  // Define a test string containing letters, numbers, spaces, and other characters  String test = "Aa kiu, I swd skieo 236587. GH kiu: sieo?? 25.33";    // Call the count method to analyze the characters in the test string  count(test);  }  public static void count(String x) {  // Convert the input string to a character array  char[] ch = x.toCharArray();    // Initialize counters for letters, spaces, numbers, and other characters  int letter = 0;  int space = 0;  int num = 0;  int other = 0;    // Iterate through the character array to categorize characters  for (int i = 0; i < x.length(); i++) {  // Check if the character is a letter  if (Character.isLetter(ch[i])) {  letter++;  }  // Check if the character is a digit (number)  else if (Character.isDigit(ch[i])) {  num++;  }  // Check if the character is a space  else if (Character.isSpaceChar(ch[i])) {  space++;  }  // Character falls into the "other" category if none of the above conditions are met  else {  other++;  }  }    // Display the original string  System.out.println("The string is : Aa kiu, I swd skieo 236587. GH kiu: sieo?? 25.33");    // Display the counts of letters, spaces, numbers, and other characters  System.out.println("letter: " + letter);  System.out.println("space: " + space);  System.out.println("number: " + num);  System.out.println("other: " + other);  }  } |

Output:

A close-up of a computer screen

Description automatically generated

**39.** Write a Java program to create and display a unique three-digit number using 1, 2, 3, 4. Also count how many three-digit numbers are there.  
*Expected Output*

123

124

...

431

432

Total number of the three-digit-number is 24

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.Scanner;  public class Exercise39 {  public static void main(String[] args) {  int amount = 0; // Initialize a counter for three-digit numbers    // Iterate through three nested loops to generate all unique three-digit numbers  for (int i = 1; i <= 4; i++) {  for (int j = 1; j <= 4; j++) {  for (int k = 1; k <= 4; k++) {  // Check if 'i,' 'j,' and 'k' are all different (not equal)  if (k != i && k != j && i != j) {  amount++; // Increment the counter  System.out.println(i + "" + j + "" + k); // Print the generated three-digit number  }  }  }  }    // Display the total number of generated three-digit numbers  System.out.println("Total number of the three-digit-number is " + amount);  }  } |

Output:

A white background with blue text

Description automatically generated

**40.** Write a Java program to list the available character sets in charset objects.  
*Expected Output*

List of available character sets:

Big5

Big5-HKSCS

CESU-8

EUC-JP

EUC-KR

GB18030

GB2312

GBK

...

x-SJIS\_0213

x-UTF-16LE-BOM

X-UTF-32BE-BOM

X-UTF-32LE-BOM

x-windows-50220

x-windows-50221

x-windows-874

x-windows-949

x-windows-950

x-windows-iso2022jp

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.nio.charset.Charset;  public class Exercise40 {  public static void main(String[] args) {  System.out.println("List of available character sets: ");    // Iterate through the available character sets and print their names  for (String str : Charset.availableCharsets().keySet()) {  System.out.println(str);  }  }  } |

Output:

A screen shot of a computer

Description automatically generated

**41.** Write a Java program to print the ASCII value of a given character.  
*Expected Output*

The ASCII value of Z is :90

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| public class Exercise41 {  public static void main(String[] args) {  // Define an integer variable 'chr' and assign the ASCII value of 'Z' to it.  int chr = 'Z';    // Print the ASCII value of 'Z' to the console.  System.out.println("The ASCII value of Z is: " + chr);  }  } |

Output:



**42.** Write a Java program to input and display your password.  
*Expected Output*

Input your Password:

Your password was: abc@123

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.io.Console;  public class Example42 {  public static void main(String[] args) {  // Declare a Console variable 'cons'.  Console cons;  // Check if the system console is available.  if ((cons = System.console()) != null) {  // Declare a character array 'pass\_ward' to store the password.  char[] pass\_ward = null;  try {  // Prompt the user to input their password.  pass\_ward = cons.readPassword("Input your Password:");  // Display the password to the console.  System.out.println("Your password was: " + new String(pass\_ward));  } finally {  // Ensure that the password array is securely cleared.  if (pass\_ward != null) {  java.util.Arrays.fill(pass\_ward, ' ');  }  }  } else {  // If the system console is not available, throw a runtime exception.  throw new RuntimeException("Can't get the password... No console");  }  }  } |

Output:

**43.** Write a Java program to print the following string in a specific format (see output).  
*Sample Output*

Twinkle, twinkle, little star,

How I wonder what you are!

Up above the world so high,

Like a diamond in the sky.

Twinkle, twinkle, little star,

How I wonder what you are

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| public class Example43 {  public static void main(String[] args) {  // Display a pattern of a nursery rhyme.  System.out.println("\nTwinkle, twinkle, little star, \n\tHow I wonder what you are! \n\t\tUp above the world so high, \n\t\tLike a diamond in the sky. \nTwinkle, twinkle, little star, \n\tHow I wonder what you are!\n\n");  }  } |

Output:

A white background with blue text

Description automatically generated

**44.** Write a Java program that accepts an integer (n) and computes the value of n+nn+nnn.  
*Sample Output:*

Input number: 5

5 + 55 + 555

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.Scanner;  public class Exercise44 {  public static void main(String[] args) {  int n;  char s1, s2, s3;  Scanner in = new Scanner(System.in);    // Prompt the user to input a number.  System.out.print("Input number: ");  n = in.nextInt();    // Display the number in a specific pattern.  System.out.printf("%d + %d%d + %d%d%d\n", n, n, n, n, n, n);  }  } |

Output:

A number on a white background

Description automatically generated

**45.** Write a Java program to find the size of a specified file.  
*Sample Output:*

/home/students/abc.txt : 0 bytes

/home/students/test.txt : 0 bytes

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.Scanner;  import java.io.File;  public class Exercise45 {  public static void main(String[] args) {  // Display the file sizes in bytes for two files.  System.out.println("/home/students/abc.txt : " + new File("abc.txt").length() + " bytes");  System.out.println("/home/students/test.txt : " + new File("test.txt").length() + " bytes");  }  } |

Output:

A close-up of a text

Description automatically generated

**46.** Write a Java program to display system time.  
*Sample Output:*

Current Date time: Fri Jun 16 14:17:40 IST 2017

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| public class Exercise46 {    public static void main(String[] args) {  // Display the current date and time using System.currentTimeMillis()  System.out.format("\nCurrent Date time: %tc%n\n", System.currentTimeMillis());  }  } |

Output:



**47.** Write a Java program to display the current date and time in a specific format.  
*Sample Output:*

Now: 2017/06/16 08:52:03.066

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.text.SimpleDateFormat;  import java.util.Calendar;  import java.util.TimeZone;  public class Exercise47 {  public static void main(String args[]) {  // Create a SimpleDateFormat with a specific date and time format  SimpleDateFormat cdt = new SimpleDateFormat("yyyy/MM/dd HH:mm:ss.SSS");    // Set the time zone for the calendar to GMT  cdt.setCalendar(Calendar.getInstance(TimeZone.getTimeZone("GMT")));    // Display the current date and time in the specified format  System.out.println("\nNow: " + cdt.format(System.currentTimeMillis()));  }  } |

Output:



**48.** Write a Java program to print odd numbers from 1 to 99. Prints one number per line.  
*Sample Output:*

1

3

5

7

9

11

....

91

93

95

97

99

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  public class Exercise48 {  public static void main(String[] args) {  // Iterate through numbers from 1 to 99  for (int i = 1; i < 100; i++) {  // Check if the number is odd  if (i % 2 != 0) {  // Print the odd number  System.out.println(i);  }  }  }  } |

Output:

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**49.** Write a Java program to accept a number and check whether the number is even or not. Prints 1 if the number is even or 0 if odd.  
*Sample Output:*

Input a number: 20

1

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  public class Exercise49 {  public static void main(String[] args) {  // Create a scanner for user input  Scanner in = new Scanner(System.in);    // Prompt the user to input a number  System.out.print("Input a number: ");  int n = in.nextInt();    // Check if the number is even and print the result  if (n % 2 == 0) {  System.out.println(1); // If the number is even, print 1  } else {  System.out.println(0); // If the number is odd, print 0  }  }  } |

Output:

A number on a white background

Description automatically generated

**50.** Write a Java program to print numbers between 1 and 100 divisible by 3, 5 and both.  
*Sample Output:*

Divided by 3:

3, 6, 9, 12, 15, 18, 21, 24, 27, 30, 33, 36, 39, 42, 45, 48, 51, 54, 57

, 60, 63, 66, 69, 72, 75, 78, 81, 84, 87, 90, 93, 96, 99,

Divided by 5:

5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60, 65, 70, 75, 80, 85, 90,

95,

Divided by 3 & 5:

15, 30, 45, 60, 75, 90,

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| public class Exercise50 {  public static void main(String args[]) {  // Print numbers divided by 3  System.out.println("\nDivided by 3: ");  for (int i = 1; i < 100; i++) {  if (i % 3 == 0)  System.out.print(i + ", ");  }  // Print numbers divided by 5  System.out.println("\n\nDivided by 5: ");  for (int i = 1; i < 100; i++) {  if (i % 5 == 0)  System.out.print(i + ", ");  }  // Print numbers divided by both 3 and 5  System.out.println("\n\nDivided by 3 & 5: ");  for (int i = 1; i < 100; i++) {  if (i % 3 == 0 && i % 5 == 0)  System.out.print(i + ", ");  }  System.out.println("\n");  }  } |

Output:

A number on a white background

Description automatically generated

**51.** Write a Java program to convert a string to an integer.  
*Sample Output:*

Input a number(string): 25

The integer value is: 25

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  public class Exercise51 {  public static void main(String[] args) {  // Create a Scanner object for user input  Scanner in = new Scanner(System.in);  // Prompt the user to input a number (as a string)  System.out.print("Input a number (string): ");  // Read the input string and store it in str1  String str1 = in.nextLine();  // Parse the string as an integer  int result = Integer.parseInt(str1);  // Display the integer value  System.out.printf("The integer value is: %d", result);  // Print a new line for better formatting  System.out.printf("\n");  }  } |

Output:

A group of blue text

Description automatically generated

**52.** Write a Java program to calculate the sum of two integers and return true if the sum is equal to a third integer.  
*Sample Output:*

Input the first number : 5

Input the second number: 10

Input the third number : 15

The result is: true

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  public class Exercise52 {  public static void main(String[] args) {  // Create a Scanner object for user input  Scanner in = new Scanner(System.in);    // Prompt the user to input the first number  System.out.print("Input the first number: ");  int x = in.nextInt();  // Prompt the user to input the second number  System.out.print("Input the second number: ");  int y = in.nextInt();  // Prompt the user to input the third number  System.out.print("Input the third number: ");  int z = in.nextInt();  // Calculate the result using the sumoftwo function and display it  System.out.print("The result is: " + sumoftwo(x, y, z));  // Print a new line for better formatting  System.out.print("\n");  }    // Function to check if the sum of two numbers equals the third number  public static boolean sumoftwo(int p, int q, int r) {  return ((p + q) == r || (q + r) == p || (r + p) == q);  }  } |

Output:

A number and numbers on a white background

Description automatically generated

**53.** Write a Java program that accepts three integers from the user. It returns true if the second number is higher than the first number and the third number is larger than the second number. If "abc" is true, the second number does not need to be larger than the first number.  
*Sample Output:*

Input the first number : 5

Input the second number: 10

Input the third number : 15

The result is: true

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  public class Exercise53 {  public static void main(String[] args) {  // Create a Scanner object for user input  Scanner in = new Scanner(System.in);    // Prompt the user to input the first number  System.out.print("Input the first number: ");  int x = in.nextInt();  // Prompt the user to input the second number  System.out.print("Input the second number: ");  int y = in.nextInt();  // Prompt the user to input the third number  System.out.print("Input the third number: ");  int z = in.nextInt();  // Prompt the user to input a boolean value (true or false)  System.out.print("Input a boolean value (true/false): ");  boolean xyz = in.nextBoolean();  // Calculate the result using the test function and display it  System.out.print("The result is: " + test(x, y, z, xyz));  // Print a new line for better formatting  System.out.print("\n");  }    // Function to test conditions based on boolean input  public static boolean test(int p, int q, int r, boolean xyz) {  if (xyz)  return (r > q);  return (q > p && r > q);  }  } |

Output:

A number with numbers and a few letters

Description automatically generated with medium confidence

**54.** Write a Java program that accepts three integers from the user and returns true if two or more of them (integers) have the same rightmost digit. The integers are non-negative.  
*Sample Output:*

Input the first number : 5

Input the second number: 10

Input the third number : 15

The result is: true

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  public class Exercise54 {  public static void main(String[] args) {  // Create a Scanner object for user input  Scanner in = new Scanner(System.in);  // Prompt the user to input the first number  System.out.print("Input the first number: ");  int x = in.nextInt();  // Prompt the user to input the second number  System.out.print("Input the second number: ");  int y = in.nextInt();  // Prompt the user to input the third number  System.out.print("Input the third number: ");  int z = in.nextInt();  // Prompt the user to input a boolean value (true or false)  System.out.print("Input a boolean value (true/false): ");  boolean xyz = in.nextBoolean();  // Calculate the result using the test\_last\_digit function and display it  System.out.print("The result is: " + test\_last\_digit(x, y, z, xyz));  // Print a new line for better formatting  System.out.print("\n");  }  // Function to test if any two numbers have the same last digit based on the boolean input  public static boolean test\_last\_digit(int p, int q, int r, boolean xyz) {  return (p % 10 == q % 10) || (p % 10 == r % 10) || (q % 10 == r % 10);  }  } |

Output:

A number on a white background

Description automatically generated

**55.** Write a Java program to convert seconds to hours, minutes and seconds.  
*Sample Output:*

Input seconds: 86399

23:59:59

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  public class Main {  public static void main(String[] args) {  // Create a Scanner object for user input  Scanner in = new Scanner(System.in);  // Prompt the user to input the total seconds  System.out.print("Input seconds: ");  int seconds = in.nextInt();  // Calculate the hours, minutes, and seconds  int S = seconds % 60; // Calculate the remaining seconds  int H = seconds / 60; // Convert total seconds to minutes  int M = H % 60; // Calculate the remaining minutes  H = H / 60; // Convert total minutes to hours  // Display the time in the format HH:MM:SS  System.out.print(H + ":" + M + ":" + S);  // Print a new line for better formatting  System.out.print("\n");  }  } |

Output:

A number on a white background

Description automatically generated

**56.** Write a Java program to find the number of values in a given range divisible by a given value.  
For example x = 5, y=20 and p =3, find the number of integers within the range x..y and that are divisible by p i.e. { i :x ≤ i ≤ y, i mod p = 0 }  
*Sample Output:*

5

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  public class Exercise56 {  public static void main(String[] args) {  // Initialize variables x, y, and p  int x = 5;  int y = 20;  int p = 3;  // Call the result method and print the result  System.out.println(result(x, y, p));  }  // Define a method to calculate the result based on x, y, and p  public static int result(int x, int y, int p) {  // Check if x is divisible by p  if (x % p == 0) {  // Calculate the result and return it  return (y / p - x / p + 1);  } else {  // Calculate the result and return it  return (y / p - x / p);  }  }  } |

Output:

A white rectangular object with a black border

Description automatically generated

**57.** Write a Java program to accept an integer and count the factors of the number.  
*Sample Output:*

Input an integer: 25

3

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  public class Exercise57 {  public static void main(String[] args) {  // Create a Scanner object for user input  Scanner in = new Scanner(System.in);  System.out.print("Input an integer: ");    // Read an integer from the user  int x = in.nextInt();    // Call the result method and print the result  System.out.println(result(x));  }  // Define a method to calculate the number of divisors for a given integer  public static int result(int num) {  int ctr = 0;    // Iterate from 1 to the square root of the input number  for (int i = 1; i <= (int) Math.sqrt(num); i++) {  // Check if 'i' is a divisor, and if it's not a perfect square  if (num % i == 0 && i \* i != num) {  ctr += 2; // Increase the count by 2  } else if (i \* i == num) {  ctr++; // If 'i' is a perfect square, increase the count by 1  }  }  return ctr; // Return the total count of divisors  }  } |

Output:

A number on a white background

Description automatically generated

**58.** Write a Java program to capitalize the first letter of each word in a sentence.  
*Sample Output:*

Input a Sentence: the quick brown fox jumps over the lazy dog.

The Quick Brown Fox Jumps Over The Lazy Dog.

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  public class Exercise58 {  public static void main(String[] args) {  // Create a Scanner object for user input  Scanner in = new Scanner(System.in);  System.out.print("Input a Sentence: ");    // Read a sentence from the user  String line = in.nextLine();    // Initialize an empty string to store the result in uppercase  String upper\_case\_line = "";    // Create a Scanner to process individual words in the sentence  Scanner lineScan = new Scanner(line);    // Iterate through the words in the sentence  while (lineScan.hasNext()) {  String word = lineScan.next();    // Capitalize the first letter of each word and append it to the result  upper\_case\_line += Character.toUpperCase(word.charAt(0)) + word.substring(1) + " ";  }    // Remove trailing space and print the result in uppercase  System.out.println(upper\_case\_line.trim());  }  } |

Output:

**59.** Write a Java program to convert a string into lowercase.  
*Sample Output:*

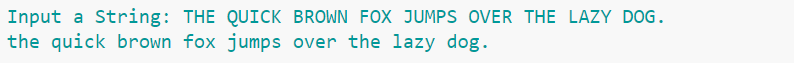
Input a String: THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG.

the quick brown fox jumps over the lazy dog.

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  public class Exercise59 {  public static void main(String[] args) {  // Create a Scanner object for user input  Scanner in = new Scanner(System.in);  System.out.print("Input a String: ");    // Read a string from the user  String line = in.nextLine();    // Convert the string to lowercase  line = line.toLowerCase();    // Print the lowercase version of the input string  System.out.println(line);  }  } |

Output:



**60.** Write a Java program to find the penultimate (next to the last) word in a sentence.  
*Sample Output:*

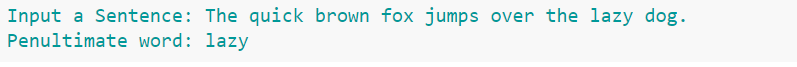
Input a String: The quick brown fox jumps over the lazy dog.

Penultimate word: lazy

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  public class Exercise60 {  public static void main(String[] args) {  // Create a Scanner object for user input  Scanner in = new Scanner(System.in);  System.out.print("Input a Sentence: ");    // Read a sentence from the user  String line = in.nextLine();    // Split the sentence into words using one or more spaces as the delimiter  String[] words = line.split("[ ]+");    // Print the penultimate word from the array  System.out.println("Penultimate word: " + words[words.length - 2]);  }  } |

Output:



**61.** Write a Java program to reverse a word.  
*Sample Output:*

Input a word: dsaf

Reverse word: fasd

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  public class Exercise61 {  public static void main(String[] args) {  // Create a Scanner object for user input  Scanner in = new Scanner(System.in);  System.out.print("\nInput a word: ");    // Read a word from the user  String word = in.nextLine();    // Trim any leading or trailing whitespaces  word = word.trim();    // Initialize an empty string for the reversed word  String result = "";    // Convert the word to a character array  char[] ch = word.toCharArray();    // Reverse the word by iterating through the characters  for (int i = ch.length - 1; i >= 0; i--) {  result += ch[i];  }    // Print the reversed word  System.out.println("Reverse word: " + result.trim());  }  } |

Output:

A white background with a black and white flag

Description automatically generated with medium confidence

**62.** Write a Java program that accepts three integer values and returns true if one is 20 or more less than the others' subtractions.  
*Sample Output:*

Input the first number : 15

Input the second number: 20

Input the third number : 25

false

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  public class Exercise62 {  public static void main(String[] args) {  // Create a Scanner object for user input  Scanner in = new Scanner(System.in);    // Prompt the user to input the first number  System.out.print("Input the first number : ");  int x = in.nextInt(); // Read and store the first number    // Prompt the user to input the second number  System.out.print("Input the second number: ");  int y = in.nextInt(); // Read and store the second number    // Prompt the user to input the third number  System.out.print("Input the third number : ");  int z = in.nextInt(); // Read and store the third number    // Calculate and print the result of the condition  // The condition checks if the absolute difference between the numbers is greater than or equal to 20  System.out.println((Math.abs(x - y) >= 20 || Math.abs(y - z) >= 20 || Math.abs(z - x) >= 20));  }  } |

Output:

A number on a white background

Description automatically generated**63.** Write a Java program that accepts two integer values from the user and returns the largest value. However if the two values are the same, return 0 and find the smallest value if the two values have the same remainder when divided by 6.  
*Sample Output:*

Input the first number : 12

Input the second number: 13

Result: 13

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  public class Exercise63 {  public static void main(String[] args) {  // Create a Scanner object for user input  Scanner in = new Scanner(System.in);  // Prompt the user to input the first number  System.out.print("Input the first number : ");  int a = in.nextInt(); // Read and store the first number    // Prompt the user to input the second number  System.out.print("Input the second number: ");  int b = in.nextInt(); // Read and store the second number  // Call the result method with the two numbers and print the result  System.out.println("Result: " + result(a, b));  }  // Define a method to calculate the result based on two input numbers  public static int result(int x, int y) {  // Check if the two numbers are equal  if (x == y) {  return 0;  }    // Check if the remainder when divided by 6 is the same for both numbers  if (x % 6 == y % 6) {  // If the remainder is the same, return the smaller number  return (x < y) ? x : y;  }    // If the remainders are different, return the larger number  return (x > y) ? x : y;  }  } |

Output:

A number on a white background

Description automatically generated**64.** Write a Java program that accepts two integer values between 25 and 75 and returns true if there is a common digit in both numbers.  
*Sample Output:*

Input the first number : 35

Input the second number: 45

Result: true

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  public class Exercise64 {  public static void main(String[] args) {  // Create a Scanner object for user input  Scanner in = new Scanner(System.in);  // Prompt the user to input the first number  System.out.print("Input the first number : ");  int a = in.nextInt(); // Read and store the first number  // Prompt the user to input the second number  System.out.print("Input the second number: ");  int b = in.nextInt(); // Read and store the second number  // Call the common\_digit method with the two numbers and print the result  System.out.println("Result: " + common\_digit(a, b));  }  // Define a method to check if there's a common digit between two numbers  public static boolean common\_digit(int p, int q) {  // Check if p is less than 25 or q is greater than 75  if (p < 25 || q > 75) {  return false;  }  // Extract the last digit of each number  int x = p % 10;  int y = q % 10;  // Remove the last digit from both numbers  p /= 10;  q /= 10;  // Check if there's a common digit by comparing the remaining digits  return (p == q || p == y || x == q || x == y);  }  } |

Output:

A number on a white background

Description automatically generated

**65.** Write a Java program to calculate the modules of two numbers without using any inbuilt modulus operator.  
*Sample Output:*

Input the first number : 19

Input the second number: 7

5

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  public class Exercise65 {  public static void main(String[] args) {  // Create a Scanner object for user input  Scanner in = new Scanner(System.in);  // Prompt the user to input the first number  System.out.print("Input the first number : ");  int a = in.nextInt(); // Read and store the first number  // Prompt the user to input the second number  System.out.print("Input the second number: ");  int b = in.nextInt(); // Read and store the second number  // Calculate the division result of a by b  int divided = a / b;  // Calculate the remainder of a divided by b  int result = a - (divided \* b);  // Print the remainder  System.out.println(result);  }  } |

Output:

A number and numbers on a white background

Description automatically generated

**66.** Write a Java program to compute the sum of the first 100 prime numbers.  
*Sample Output:*

Sum of the first 100 prime numbers: 24133

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.lang.\*;  public class Exercise67 {  public static void main(String[] args) {  // Define the main string and the word to insert  String main\_string = "Python 3.0";  String word = "Tutorial";  // Create a new string by concatenating substrings  String result = main\_string.substring(0, 7) + word + main\_string.substring(6);  // Print the resulting string  System.out.println(result);  }  } |

Output:



**67.** Write a Java program to insert a word in the middle of another string.  
Insert "Tutorial" in the middle of "Python 3.0", so the result will be Python Tutorial 3.0.  
*Sample Output:*

Python Tutorial 3.0

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.lang.\*;  public class Exercise67 {  public static void main(String[] args) {  // Define the main string and the word to insert  String main\_string = "Python 3.0";  String word = "Tutorial";  // Create a new string by concatenating substrings  String result = main\_string.substring(0, 7) + word + main\_string.substring(6);  // Print the resulting string  System.out.println(result);  }  } |

Output:

A close-up of a number

Description automatically generated

**68.** Write a Java program to create another string of 4 copies of the last 3 characters of the original string. The original string length must be 3 and above.  
*Sample Output:*

3.03.03.03.0

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.lang.\*;  public class Exercise68 {  public static void main(String[] args) {  // Define the main string  String main\_string = "Python 3.0";  // Extract the last three characters from the main string  String last\_three\_chars = main\_string.substring(main\_string.length() - 3);  // Repeat the last three characters four times and print the result  System.out.println(last\_three\_chars + last\_three\_chars + last\_three\_chars + last\_three\_chars);  }  } |

Output:

A white rectangular object with a black border

Description automatically generated

**69.** Write a Java program to extract the first half of a even string.  
Test Data: Python  
*Sample Output:*

Pyt

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.lang.\*;  public class Exercise69 {  public static void main(String[] args) {  // Define the main string  String main\_string = "Python";  // Extract the substring from the beginning to the middle of the string  String substring = main\_string.substring(0, main\_string.length() / 2);  // Print the extracted substring  System.out.println(substring);  }  } |

Output:



**70.** Write a Java program to create a string in the form of short\_string + long\_string + short\_string from two strings. The strings must not have the same length.  
Test Data: Str1 = Python  
Str2 = Tutorial  
*Sample Output:*

PythonTutorialPython

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.lang.\*;  public class Exercise70 {  public static void main(String[] args) {  // Define two strings  String str1 = "Python";  String str2 = "Tutorial";  // Check the lengths of the strings  if (str1.length() >= str2.length()) {  // Concatenate the strings in the order: str2 + str1 + str2  System.out.println(str2 + str1 + str2);  } else {  // Concatenate the strings in the order: str1 + str2 + str1  System.out.println(str1 + str2 + str1);  }  }  } |

Output:



**71.** Write a Java program to create the concatenation of the two strings except removing the first character of each string. The length of the strings must be 1 and above.  
Test Data: Str1 = Python  
Str2 = Tutorial  
*Sample Output:*

ythonutorial

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.lang.\*;  public class Exercise71 {  public static void main(String[] args) {  // Define two strings  String str1 = "Python";  String str2 = "Tutorial";  // Print the substrings of both strings, excluding the first character  System.out.println(str1.substring(1) + str2.substring(1));  }  } |

Output:



**72.** Write a Java program to create a string taking the first three characters from a given string. If the string length is less than 3 use "#" as substitute characters.  
Test Data: Str1 = " "  
*Sample Output:*

###

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.lang.\*;  public class Exercise72 {  public static void main(String[] args) {  // Define an empty string  String str1 = "";  // Get the length of the string  int len = str1.length();  // Check the length of the string and take different actions based on its length  if (len >= 3) {  // If the string has three or more characters, print the first three characters  System.out.println(str1.substring(0, 3));  } else if (len == 1) {  // If the string has only one character, add "##" to it and print  System.out.println(str1.charAt(0) + "##");  } else {  // If the string is empty or has two characters, print "###"  System.out.println("###");  }  }  } |

Output:



**73.** Write a Java program to create a string taking the first and last characters from two given strings. If the length of each string is 0 use "#" for missing characters.  
Test Data: str1 = "Python"  
str2 = " "  
*Sample Output:*

P#

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.lang.\*;  public class Exercise73 {  public static void main(String[] args) {  // Define two input strings  String str1 = "Python";  String str2 = "";  // Get the length of the second string  int length2 = str2.length();  // Initialize a result string  String result = "";  // Check the length of the first string and append the first character (if available) or '#'  result += (str1.length() >= 1) ? str1.charAt(0) : '#';  // Check the length of the second string and append the last character (if available) or '#'  result += (length2 >= 1) ? str2.charAt(length2 - 1) : '#';  // Print the resulting string  System.out.println(result);  }  } |

Output:



**74.** Write a Java program to test if 10 appears as the first or last element of an array of integers. The array length must be broader than or equal to 2.  
*Sample Output:*  
Test Data: array = 10, -20, 0, 30, 40, 60, 10

true

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.lang.\*;  public class Exercise74 {  public static void main(String[] args) {  // Define an integer array  int[] num\_array = {10, -20, 0, 30, 40, 60, 10};    // Check if the first or last element of the array is equal to 10  boolean result = (num\_array[0] == 10 || num\_array[num\_array.length - 1] == 10);  // Print the result  System.out.println(result);  }  } |

Output:



**75.** Write a Java program to test if the first and last elements of an array of integers are the same. The array length must be broader than or equal to 2.  
Test Data: array = 50, -20, 0, 30, 40, 60, 10  
*Sample Output:*

false

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.lang.\*;  public class Exercise75 {  public static void main(String[] args) {  // Define an integer array  int[] num\_array = {50, -20, 0, 30, 40, 60, 10};    // Check if the array has at least 2 elements and if the first and last elements are equal  boolean result = (num\_array.length >= 2 && num\_array[0] == num\_array[num\_array.length - 1]);  // Print the result  System.out.println(result);  }  } |

Output:



**76.** Write a Java program to test if the first and last element of two integer arrays are the same. The array length must be greater than or equal to 2.  
Test Data: array1 = 50, -20, 0, 30, 40, 60, 12  
array2 = 45, 20, 10, 20, 30, 50, 11  
*Sample Output:*

false

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| public class Main {  public static void main(String[] args) {  // Define two integer arrays (num\_array1 and num\_array2)    // Example 1: Arrays with different first and last elements  int[] num\_array1 = {50, -20, 0, 30, 40, 60, 12};  int[] num\_array2 = {45, 20, 10, 20, 30, 50, 11};    // Example 2: Arrays with the same first and last elements  // int[] num\_array1 = {50, -20, 0, 30, 40, 60, 50};  // int[] num\_array2 = {45, 20, 10, 20, 30, 50, 45};    // Example 3: Arrays with lengths less than 2  // int[] num\_array1 = {50};  // int[] num\_array2 = {45};    if (num\_array1.length >= 2 && num\_array2.length >= 2) {  // Check if both arrays have lengths of at least 2 and if their first or last elements are equal  System.out.println(num\_array1[0] == num\_array2[0] || num\_array1[num\_array1.length - 1] == num\_array2[num\_array2.length - 1]);  } else {  System.out.println("Array lengths less than 2.");  }  }  } |

Output:

A white background with black and white clouds

Description automatically generated with medium confidence

**77.** Write a Java program to create an array of length 2 from two integer arrays with three elements. The newly created array will contain the first and last elements from the two arrays.  
Test Data: array1 = 50, -20, 0  
array2 = 5, -50, 10  
*Sample Output:*

Array1: [50, -20, 0]

Array2: [5, -50, 10]

New Array: [50, 10]

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.Arrays;  public class Exercise77 {  public static void main(String[] args) {  // Define two integer arrays (array1 and array2)  int[] array1 = {50, -20, 0};  int[] array2 = {5, -50, 10};    // Print the elements of array1  System.out.println("Array1: " + Arrays.toString(array1));    // Print the elements of array2  System.out.println("Array2: " + Arrays.toString(array2));    // Create a new array, array\_new, with elements from array1 and array2  int[] array\_new = {array1[0], array2[2]};    // Print the elements of the new array, array\_new  System.out.println("New Array: " + Arrays.toString(array\_new));  }  } |

Output:

A white background with black dots

Description automatically generated

**78.** Write a Java program to test that a given array of integers of length 2 contains a 4 or a 7.  
*Sample Output:*

Original Array: [5, 7]

true

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.Arrays;  public class Exercise78 {  public static void main(String[] args) {  // Define an integer array, array\_nums  int[] array\_nums = {5, 7};    // Print the elements of the original array  System.out.println("Original Array: " + Arrays.toString(array\_nums));    // Check if the first element of the array is 4 or 7 and print the result  if (array\_nums[0] == 4 || array\_nums[0] == 7) {  System.out.println("True");  } else {  // If the first element is not 4 or 7, check the second element and print the result  System.out.println(array\_nums[1] == 4 || array\_nums[1] == 7);  }  }  } |

Output:

A number on a white background

Description automatically generated

**79.** Write a Java program to rotate an array (length 3) of integers in the left direction.  
*Sample Output:*

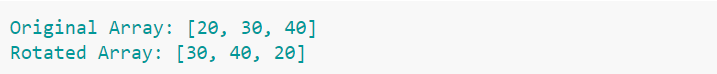
Original Array: [20, 30, 40]

Rotated Array: [30, 40, 20]

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.Arrays;  public class Exercise79 {  public static void main(String[] args) {  // Define an integer array, array\_nums  int[] array\_nums = {20, 30, 40};    // Print the elements of the original array  System.out.println("Original Array: " + Arrays.toString(array\_nums));    // Create a new array by rotating the elements of the original array  int[] new\_array\_nums = {array\_nums[1], array\_nums[2], array\_nums[0]};    // Print the elements of the rotated array  System.out.println("Rotated Array: " + Arrays.toString(new\_array\_nums));  }  } |

Output:

**80.** Write a Java program to get the largest value between the first and last elements of an array (length 3) of integers.  
*Sample Output:*

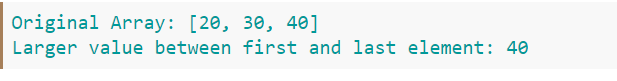
Original Array: [20, 30, 40]

Larger value between first and last element: 40

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.Arrays;  public class Exercise80 {  public static void main(String[] args) {  // Define an integer array, array\_nums  int[] array\_nums = {20, 30, 40};    // Print the elements of the original array  System.out.println("Original Array: " + Arrays.toString(array\_nums));    // Initialize a variable, max\_val, with the first element of the array  int max\_val = array\_nums[0];    // Check if the last element of the array is greater than max\_val and update max\_val if necessary  if (array\_nums[2] >= max\_val)  max\_val = array\_nums[2];    // Print the larger value between the first and last elements of the array  System.out.println("Larger value between first and last element: " + max\_val);  }  } |

Output:



**81.** Write a Java program to swap the first and last elements of an array (length must be at least 1) and create another array.  
*Sample Output:*

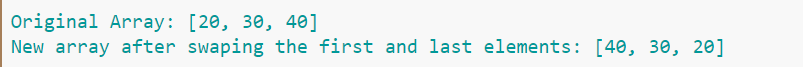
Original Array: [20, 30, 40]

New array after swaping the first and last elements: [40, 30, 20]

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.Arrays;  public class Exercise81 {  public static void main(String[] args) {  // Define an integer array, array\_nums  int[] array\_nums = {20, 30, 40};    // Print the elements of the original array  System.out.println("Original Array: " + Arrays.toString(array\_nums));    // Store the value of the first element in the variable x  int x = array\_nums[0];    // Swap the first and last elements of the array  array\_nums[0] = array\_nums[array\_nums.length - 1];  array\_nums[array\_nums.length - 1] = x;    // Print the new array after swapping the first and last elements  System.out.println("New array after swapping the first and last elements: " + Arrays.toString(array\_nums));  }  } |

Output:

**82.** Write a Java program to find the largest element between the first, last, and middle values in an array of integers (even length).  
*Sample Output:*

Original Array: [20, 30, 40, 50, 67]

Largest element between first, last, and middle values: 67

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.Arrays;  public class Exercise82 {  public static void main(String[] args) {  // Define an integer array, array\_nums  int[] array\_nums = {20, 30, 40, 50, 67};    // Print the elements of the original array  System.out.println("Original Array: " + Arrays.toString(array\_nums));    // Initialize a variable max\_val with the value of the first element  int max\_val = array\_nums[0];    // Check if the last element is greater than max\_val  if (max\_val <= array\_nums[array\_nums.length - 1])  max\_val = array\_nums[array\_nums.length - 1];    // Check if the middle element is greater than max\_val  if (max\_val <= array\_nums[array\_nums.length / 2])  max\_val = array\_nums[array\_nums.length / 2];    // Print the largest element among the first, last, and middle values  System.out.println("Largest element between first, last, and middle values: " + max\_val);  }  } |

Output:



**83.** Write a Java program to multiply the corresponding elements of two integer arrays.  
*Sample Output:*

Array1: [1, 3, -5, 4]

Array2: [1, 4, -5, -2]

Result: 1 12 25 -8

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  public class Exercise83 {  public static void main(String[] args) {  // Initialize a string to store the result  String result = "";    // Define two integer arrays  int[] left\_array = {1, 3, -5, 4};  int[] right\_array = {1, 4, -5, -2};    // Print the elements of Array1  System.out.println("\nArray1: " + Arrays.toString(left\_array));    // Print the elements of Array2  System.out.println("\nArray2: " + Arrays.toString(right\_array));    // Multiply corresponding elements from both arrays and build the result string  for (int i = 0; i < left\_array.length; i++) {  int num1 = left\_array[i];  int num2 = right\_array[i];  result += Integer.toString(num1 \* num2) + " ";  }    // Print the result string  System.out.println("\nResult: " + result);  }  } |

Output:

A white background with black numbers

Description automatically generated**84.** Write a Java program to take the last three characters from a given string. It will add the three characters at both the front and back of the string. String length must be greater than three and more.  
Test data: "Python" will be "honPythonhon"  
*Sample Output:*

honPythonhon

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  import java.io.\*;  public class Exercise84 {  public static void main(String[] args) {  // Initialize a string  String string1 = "Python";    // Define a subpart length  int slength = 3;    // Check if the subpart length is greater than the string length  if (slength > string1.length()) {  slength = string1.length(); // Limit it to the string length  }  // Get the subpart of the string from the last 3 characters  String subpart = string1.substring(string1.length() - 3);    // Print the result, which is the subpart followed by the original string and the subpart again  System.out.println(subpart + string1 + subpart);  }  } |

Output:



**85.** Write a Java program to check if a string starts with a specified word.  
Sample Data: string1 = "Hello how are you?"  
*Sample Output:*

true

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  import java.io.\*;  public class Exercise85 {  public static void main(String[] args) {  // Initialize a string  String string1 = "Hello how are you?";    // Check if the string starts with "Hello"  boolean startsWithHello = string1.startsWith("Hello");    // Print the result indicating whether the string starts with "Hello"  System.out.println(startsWithHello);  }  } |

Output:

**86.** Write a Java program starting with an integer n, divide it by 2 if it is even, or multiply it by 3 and add 1 if it is odd. Repeat the process until n = 1.

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.Scanner;  public class Main {  public static void main(String[] args) {  // Prompt the user to input the value of n  System.out.println("Input the value of n: ");    // Create a Scanner object to read user input  Scanner in = new Scanner(System.in);    // Read an integer from the user  int n = in.nextInt();    // Continue looping until n becomes 1  while (n != 1) {  // Check if n is even  if (n % 2 == 0) {  n = n / 2; // If even, divide n by 2  } else {  n = (3 \* n + 1) / 2; // If odd, perform a calculation  }  }    // Print the final value of n  System.out.println("\nValue of n = " + n);    // Close the Scanner  in.close();  }  } |

Output:

A white background with a black and white flag

Description automatically generated with medium confidence

**87.** Write a Java program that then reads an integer and calculates the sum of its digits and writes the number of each digit of the sum in English.

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.io.\*;  public class Main {  public static void main(String[] args) {  // Create a BufferedReader to read input from the user  BufferedReader br = new BufferedReader(new InputStreamReader(System.in));  try {  int sum = 0;  // Read a line of text from the user  String str = br.readLine();  char[] numStr = str.toCharArray();  // Calculate the sum of the digits in the input number  for (int i = 0; i < numStr.length; i++) {  sum += numStr[i] - '0';  }  // Print the original number and call the print\_number function  System.out.println("Original Number: " + str);  print\_number(sum);  } catch (IOException e) {  e.printStackTrace();  }  }  public static void print\_number(int n) {  int x;  int y;  int z;  String[] number = {"zero", "one", "two", "three", "four", "five", "six", "seven", "eight", "nine"};  // Print the sum of the digits of the number  System.out.println("Sum of the digits of the said number: " + n);  if (n < 10) {  // If the number is less than 10, print the corresponding word  System.out.println(number[n]);  } else if (n < 100) {  // If the number is less than 100, split it into tens and ones  x = n / 10;  y = n - x \* 10;  System.out.println("In English: " + number[x] + " " + number[y]);  } else {  // If the number is three digits, split it into hundreds, tens, and ones  x = n / 100;  y = (n - x \* 100) / 10;  z = n - x \* 100 - y \* 10;  System.out.println("In English: " + number[x] + " " + number[y] + " " + number[z]);  }  }  } |

Output:

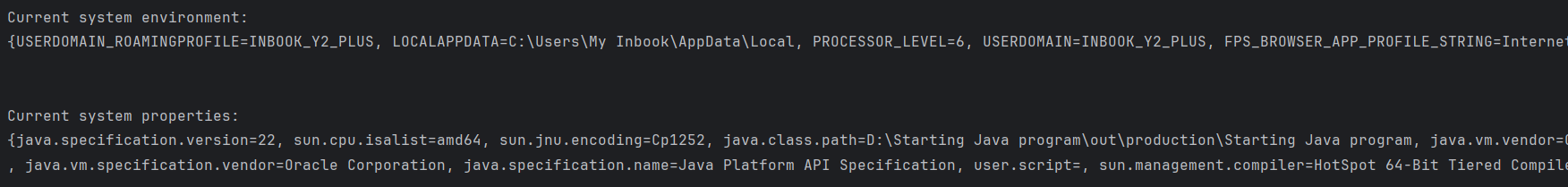
A close-up of a white background

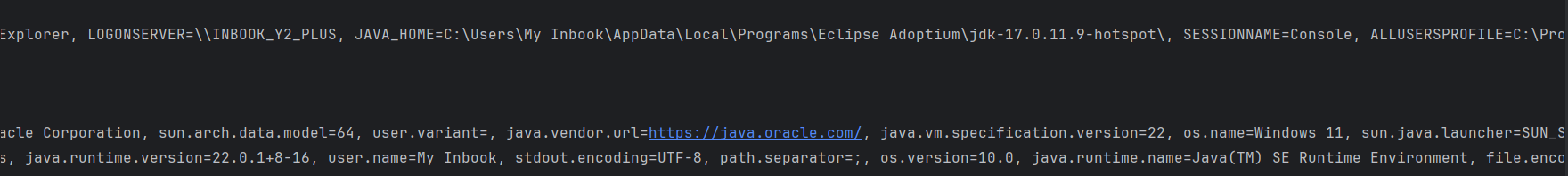
Description automatically generated

**88.** Write a Java program to get the current system environment and system properties.  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.lang.\*;  public class Exercise88 {  public static void main(String[] args) {  // Print a message indicating the display of the current system environment  System.out.println("\nCurrent system environment:");  // Retrieve and print the system environment variables using System.getenv()  System.out.println(System.getenv());  // Print a message indicating the display of the current system properties  System.out.println("\n\nCurrent system properties:");  // Retrieve and print the system properties using System.getProperties()  System.out.println(System.getProperties());  }  } |

Output:





**89.** Write a Java program to check whether a security manager has already been established for the current application or not.  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.lang.\*;  public class Exercise89 {  public static void main(String[] args) {  // Print a message indicating the display of the system security interface  System.out.println("System security interface:");  // Retrieve and print the system security manager using System.getSecurityManager()  System.out.println(System.getSecurityManager());  }  } |

Output:

A close-up of a blue dot

Description automatically generated**90.** Write a Java program to get the value of environment variables PATH, TEMP, USERNAME.  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.lang.\*;  public class Exercise90 {  public static void main(String[] args) {  // Display the value of the specified environment variable "PATH"  System.out.println("\nEnvironment variable PATH: ");  System.out.println(System.getenv("PATH"));  // Display the value of the specified environment variable "TEMP"  System.out.println("\nEnvironment variable TEMP: ");  System.out.println(System.getenv("TEMP"));  // Display the value of the specified environment variable "USERNAME"  System.out.println("\nEnvironment variable USERNAME: ");  System.out.println(System.getenv("USERNAME"));  }  } |

Output:

A white screen with blue text

Description automatically generated**91.** Write a Java program to measure how long code executes in nanoseconds.  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.lang.\*;  public class Exercise91 {  public static void main(String[] args) {  long startTime = System.nanoTime();    // Sample program: Display the first 10 natural numbers.  int i;  System.out.println("The first 10 natural numbers are:\n");  for (i = 1; i <= 10; i++) {  System.out.println(i);  }    // Calculate the elapsed time in nanoseconds.  long estimatedTime = System.nanoTime() - startTime;  System.out.println("Estimated time (in nanoseconds) to get the first 10 natural numbers: " + estimatedTime);  }  } |

Output:

A screenshot of a computer

Description automatically generated

**92.** Write a Java program to count the number of even and odd elements in a given array of integers.  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  public class Exercise92 {  public static void main(String[] args) {  // Initialize an array of integers  int[] nums = {5, 7, 2, 4, 9};    // Initialize counters for even and odd numbers  int ctr\_even = 0, ctr\_odd = 0;    // Display the original array  System.out.println("Original Array: " + Arrays.toString(nums));  // Iterate through the array to count even and odd numbers  for (int i = 0; i < nums.length; i++) {  if (nums[i] % 2 == 0) {  // Increment the counter for even numbers  ctr\_even++;  } else {  // Increment the counter for odd numbers  ctr\_odd++;  }  }    // Print the counts of even and odd elements in the array  System.out.printf("\nNumber of even elements in the array: %d", ctr\_even);  System.out.printf("\nNumber of odd elements in the array: %d", ctr\_odd);  System.out.printf("\n");  }  } |

Output:

A close up of text

Description automatically generated

**93.** Write a Java program to test if an array of integers contains an element 10 next to 10 or an element 20 next to 20, but not both.  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  public class Exercise93 {  public static void main(String[] args) {  // Define an array of integers  int[] nums = {10, 10, 2, 4, 20, 20};    // Initialize counters for even and odd numbers  int ctr\_even = 0, ctr\_odd = 0;    // Display the original array  System.out.println("Original Array: " + Arrays.toString(nums));  // Initialize boolean variables to check for specific patterns  boolean found1010 = false;  boolean found2020 = false;    // Iterate through the array to find patterns (e.g., 1010 and 2020)  for (int i = 0; i < nums.length - 1; i++) {  if (nums[i] == 10 && nums[i + 1] == 10) {  found1010 = true;  }  if (nums[i] == 20 && nums[i + 1] == 20) {  found2020 = true;  }  }  // Check if the patterns 1010 and 2020 were found and print the result  System.out.printf(String.valueOf(found1010 != found2020));  System.out.printf("\n");  }  } |

Output:

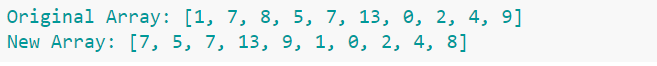
A number on a white background

Description automatically generated

**94.** Write a Java program to rearrange all the elements of a given array of integers so that all the odd numbers come before all the even numbers.  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  public class Exercise94 {  public static void main(String[] args) {  // Define an array of integers  int[] array\_nums = {1, 7, 8, 5, 7, 13, 0, 2, 4, 9};    // Initialize an index variable to track even numbers  int i = 0;  // Display the original array  System.out.println("Original Array: " + Arrays.toString(array\_nums));  // Find the first odd number's index (i)  while (i < array\_nums.length && array\_nums[i] % 2 == 0)  i++;    // Rearrange the array such that odd numbers come after even numbers  for (int j = i + 1; j < array\_nums.length; j++) {  if (array\_nums[j] % 2 != 0) {  // Swap even and odd numbers  int temp = array\_nums[i];  array\_nums[i] = array\_nums[j];  array\_nums[j] = temp;  i++;  }  }  // Display the modified array  System.out.println("New Array: " + Arrays.toString(array\_nums));  }  } |

Output:

**95.** Write a Java program to create an array (length # 0) of string values. The elements will contain "0", "1", "2" … through ... n-1.  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  public class Exercise95 {  public static void main(String[] args) {  // Define the number of elements in the array  int n = 5;    // Create an array of strings with n elements  String[] arr\_string = new String[n];  // Initialize the array with string representations of numbers  for (int i = 0; i < n; i++) {  arr\_string[i] = String.valueOf(i);  }  // Display the new array  System.out.println("New Array: " + Arrays.toString(arr\_string));  }  } |

Output:



**96.** Write a Java program to check if there is a 10 in an array of integers with a 20 somewhere later on.  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  public class Exercise96 {  public static void main(String[] args) {  int[] array\_nums = {10, 70, 80, 50, 20, 13, 50};  boolean testd = false; // Initialize a boolean variable for testing  int result = 0; // Initialize a result variable to track the outcome  int x = 10; // Define the first number to search for  int y = 20; // Define the second number to search for  // Iterate through the array  for (int i = 0; i < array\_nums.length; i++) {  if (array\_nums[i] == x)  testd = true; // Set the boolean flag when x is found in the array  // Check if both x and y are found in the array with specific conditions  if (testd && array\_nums[i] == y) {  System.out.printf(String.valueOf(true)); // Print true when both x and y are found  result = 1; // Set the result to 1  }  }  // If result is still 0, it means y did not follow x in the array  if (result == 0) {  System.out.printf(String.valueOf(false)); // Print false  }  System.out.printf("\n"); // Add a newline character for formatting  }  } |

Output:



**97.** Write a Java program to check if an array of integers contains a specified number next to each other or two same numbers separated by one element.  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  public class Exercise97 {  public static void main(String[] args) {  int[] array\_nums = {10, 20, 10, 50, 20, 13, 50};  //int[] array\_nums = {10, 10, 50, 50, 20, 13, 50};  boolean testd = false; // Initialize a boolean variable for testing  int result = 0; // Initialize a result variable to track the outcome  int x = 10; // Define the number to search for  // Iterate through the array, considering pairs of adjacent elements  for (int i = 0; i < array\_nums.length - 1; i++) {  if (array\_nums[i] == x && array\_nums[i + 1] == x) {  System.out.printf(String.valueOf(true)); // Print true when consecutive x elements are found  result = 1; // Set the result to 1  }  // Check if the element x is followed by another x with one element in between  if (i <= array\_nums.length - 3 && array\_nums[i] == x && array\_nums[i + 2] == x) {  System.out.printf(String.valueOf(true)); // Print true when x elements are found with one element in between  result = 1; // Set the result to 1  }  }  // If result is still 0, it means the specific pattern was not found  if (result == 0) {  System.out.printf(String.valueOf(false)); // Print false  }  System.out.printf("\n"); // Add a newline character for formatting  }  } |

Output:

A white rectangular object with a black border

Description automatically generated

**98.** Write a Java program to check if the value 20 appears three times and no 20's are next to each other in the array of integers.  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  public class Exercise98 {  public static void main(String[] args) {  int[] array\_nums = {10, 20, 10, 20, 40, 13, 20};  int count = 0; // Initialize a count variable to keep track of the number of occurrences of 20  int result = 0; // Initialize a result variable to track the outcome  // Check if the array has at least one element and if the first element is 20  if (array\_nums.length >= 1 && array\_nums[0] == 20) {  count++; // Increment the count if the first element is 20  }  // Iterate through the array, starting from the second element (index 1)  for (int i = 1; i < array\_nums.length; i++) {  // Check if the previous element and the current element are both 20  if (array\_nums[i - 1] == 20 && array\_nums[i] == 20) {  System.out.printf(String.valueOf(false)); // Print false when consecutive 20 elements are found  result = 1; // Set the result to 1  }  // Check if the current element is 20 and increment the count  if (array\_nums[i] == 20) {  count++;  }  }  // If result is still 0, it means the specific pattern was not found  if (result == 0) {  System.out.printf(String.valueOf(count == 3)); // Print true if count is equal to 3  }  System.out.printf("\n"); // Add a newline character for formatting  }  } |

Output:

A white background with black text

Description automatically generated with medium confidence**99.** Write a Java program that checks if a specified number appears in every pair of adjacent integers of a given array of integers.  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  public class Exercise99 {  public static void main(String[] args) {  int[] array\_nums = {10, 20, 10, 20, 40, 20, 50};  int result = 0; // Initialize a result variable  int x = 20; // The value to check for  // Iterate through the array, stopping at the second-to-last element  for (int i = 0; i < array\_nums.length - 1; i++) {  // Check if the current element and the next element are not equal to the value 'x'  if (array\_nums[i] != x && array\_nums[i + 1] != x) {  result = 1; // If the condition is met, set the result to 1  }  }  // If result is still 0, it means no adjacent pairs without 'x' were found  if (result == 0) {  System.out.printf(String.valueOf(true)); // Print true  } else {  System.out.printf(String.valueOf(false)); // Print false  }  }  } |

Output:

A white rectangular object with a black border

Description automatically generated

**100.** Write a Java program to count the elements that differ by 1 or less between two given arrays of integers with the same length.  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  public class Exercise100 {  public static void main(String[] args) {  int[] array\_nums1 = {10, 11, 10, 20, 43, 20, 50};  int[] array\_nums2 = {10, 13, 11, 20, 44, 30, 50};  System.out.println("Array1: "+Arrays.toString(array\_nums1));  System.out.println("Array2: "+Arrays.toString(array\_nums2));    int ctr = 0; // Initialize a counter to keep track of the number of elements  // Iterate through the arrays to compare elements at the same index  for (int i = 0; i < array\_nums1.length; i++) {  // Check if the absolute difference between elements is less than or equal to 1  // and the elements are not equal  if (Math.abs(array\_nums1[i] - array\_nums2[i]) <= 1 && array\_nums1[i] != array\_nums2[i]) {  ctr++; // If the condition is met, increment the counter  }  }    System.out.printf("Number of elements: "+ctr); // Print the number of elements meeting the condition  System.out.printf("\n");  }  } |

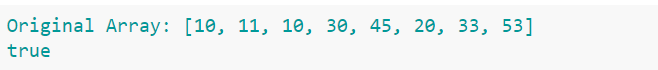
Output:

A number with blue text

Description automatically generated with medium confidence**101.** Write a Java program to determine whether the number 10 in a given array of integers exceeds 20.  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

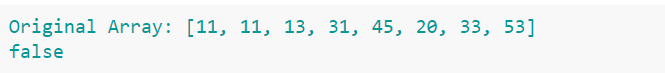
|  |
| --- |
| import java.util.\*;  public class Exercise101 {  public static void main(String[] args) {  int[] array\_nums = {10, 11, 10, 30, 45, 20, 33, 53};  int result = 0;  System.out.println("Original Array: "+Arrays.toString(array\_nums));  int ctr1 = 0; // Initialize a counter to count occurrences of value 10  int ctr2 = 0; // Initialize a counter to count occurrences of value 20  for(int i = 0; i < array\_nums.length; i++) {  if(array\_nums[i] == 10)  ctr1++; // Increment ctr1 when the element is equal to 10  if(array\_nums[i] == 20)  ctr2++; // Increment ctr2 when the element is equal to 20  }  System.out.printf(String.valueOf(ctr1 > ctr2)); // Check if the count of 10 is greater than the count of 20  System.out.printf("\n");  }  } |

Output:

**102.** Write a Java program to check if a specified array of integers contains 10 or 30.  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  public class Exercise102 {  public static void main(String[] args) {  int[] array\_nums = {11, 11, 13, 31, 45, 20, 33, 53};  int result = 1;  System.out.println("Original Array: "+Arrays.toString(array\_nums));  // Iterate through the array  for(int i = 0; i < array\_nums.length; i++) {  if(array\_nums[i] == 10 || array\_nums[i] == 30) {  // If the element is equal to 10 or 30, set the result to 0  result = 0;  }  }  if (result == 1)  System.out.printf(String.valueOf(false)); // If the result is 1, print false  else  System.out.printf(String.valueOf(true)); // If the result is not 1, print true  }  } |

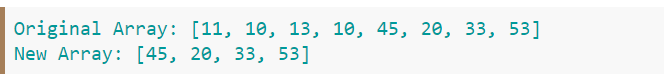
Output:



**103.** Write a Java program to create an array from a given array of integers. The newly created array will contain elements from the given array after the last element value is 10.  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  public class Exercise103 {  public static void main(String[] args) {  int[] array\_nums = {11, 10, 13, 10, 45, 20, 33, 53};  int result = 0;  System.out.println("Original Array: "+Arrays.toString(array\_nums));  int l = array\_nums.length - 1;  int[] new\_array;  // Find the last occurrence of 10 in the array  while(array\_nums[l] != 10)  l--;  // Create a new array with elements after the last 10  new\_array = new int[array\_nums.length - 1 - l];  for(int i = l + 1; i < array\_nums.length; i++)  new\_array[i - l - 1] = array\_nums[i];  System.out.println("New Array: "+Arrays.toString(new\_array));  }  } |

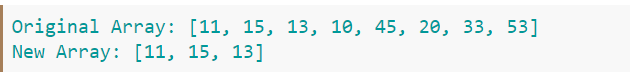
Output:



**104.** Write a Java program to create an array from a given array of integers. The newly created array will contain the elements from the given array before the last element value of 10.  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  public class Exercise104 {  public static void main(String[] args) {  int[] array\_nums = {11, 15, 13, 10, 45, 20, 33, 53};  int result = 0;  System.out.println("Original Array: "+Arrays.toString(array\_nums));    int l = 0;  int[] new\_array;  // Find the first occurrence of 10 in the array  while(array\_nums[l] != 10)  l++;  // Create a new array with elements before the first occurrence of 10  new\_array = new int[l];  for(int i = 0; i < l; i++)  new\_array[i] = array\_nums[i];  System.out.println("New Array: "+Arrays.toString(new\_array));  }  } |

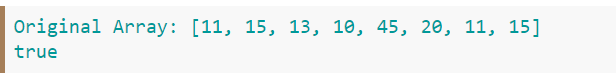
Output:



**105.** Write a Java program to check if a group of numbers (l) at the start and end of a given array are the same.  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  public class Exercise105 {  public static void main(String[] args) {  int[] array\_nums = {11, 15, 13, 10, 45, 20, 11, 15};  System.out.println("Original Array: " + Arrays.toString(array\_nums));  int result = 0;  int l = 2;  int start = 0;  int end = array\_nums.length - l;    // Check if the elements at corresponding positions within the range [start, end] are equal.  for (; l > 0; l--) {  if (array\_nums[start] != array\_nums[end]) {  result = 1;  } else {  start++;  end++;  }  }  if (result == 1) {  System.out.printf(String.valueOf(false));  } else {  System.out.printf(String.valueOf(true));  }  System.out.printf("\n");  }  } |

Output:



**106.** Write a Java program to create an array left shifted from a given array of integers.  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  public class Exercise106 {  public static void main(String[] args) {  int[] array\_nums = {11, 15, 13, 10, 45, 20};  System.out.println("Original Array: "+Arrays.toString(array\_nums));    if (array\_nums.length > 1) {  int first = array\_nums[0];  // Shift elements to the left by one position  for (int i = 1; i < array\_nums.length; i++)  array\_nums[i - 1] = array\_nums[i];  // Move the first element to the end of the array  array\_nums[array\_nums.length - 1] = first;  System.out.println("New Array: "+Arrays.toString(array\_nums));  }  }  } |

Output:

A number with blue numbers

Description automatically generated with medium confidence

**107.** Write a Java program to check if an array of integers contains three increasing adjacent numbers.  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  public class Exercise107 {  public static void main(String[] args) {  // Initialize an integer array with some values  int[] array\_nums = {11, 12, 13, 14, 45, 20};  // Print the original array  System.out.println("Original Array: " + Arrays.toString(array\_nums));  // Initialize a result variable to 1  int result = 1;  // Iterate through the array to check for consecutive integers  for (int i = 0; i <= array\_nums.length - 3; i++) {  // Check if the current number, the next, and the one after that are consecutive  if (array\_nums[i] + 1 == array\_nums[i + 1] && array\_nums[i + 1] + 1 == array\_nums[i + 2]) {  // If they are consecutive, set the result to 0  result = 0;  }  }  // Check the result and print either true or false  if (result == 1) {  System.out.printf(String.valueOf(false));  } else {  System.out.printf(String.valueOf(true));  }  // Print a newline character for formatting  System.out.printf("\n");  }  } |

Output:

A number on a white background

Description automatically generated

**108.** Write a Java program to add all the digits of a given positive integer until the result has a single digit.  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.Scanner;  public class Example108 {  public static void main(String[] arg) {  // Create a Scanner object for user input  Scanner in = new Scanner(System.in);  // Prompt the user to input a positive integer  System.out.print("Input a positive integer: ");    // Read the user's input as an integer  int n = in.nextInt();  if (n > 0) {  // Check if n is a positive integer  System.out.print("The single digit number is: " + (n == 0 ? 0 : (n % 9 == 0 ? 9 : n % 9)));  }    // Close the input scanner  in.close();  System.out.println("\n");  }  } |

Output:

A white background with blue text

Description automatically generated

**109.** Write a Java program to form a staircase shape of n coins where every k-th row must have exactly k coins.  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.Scanner;  public class Example109 {  public static void main(String[] arg) {  // Create a Scanner object for user input  Scanner in = new Scanner(System.in);  // Prompt the user to input a positive integer  System.out.print("Input a positive integer: ");    // Read the user's input as an integer  int n = in.nextInt();  if (n > 0) {  // Check if n is a positive integer  double sqrtResult = Math.sqrt(8 \* (long) n + 1); // Calculate the square root  int numRows = (int) ((sqrtResult - 1) / 2); // Calculate the number of rows  System.out.println("Number of rows: " + numRows);  }    // Close the input scanner  in.close();  }  } |

Output:

A close-up of a number

Description automatically generated

**110.** Write a Java program to check whether the given integer is a power of 4 or not.  
Given num = 64, return true. Given num = 6, return false.  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.Scanner;  public class Example110 {  public static void main(String[] arg) {  int test = 0; // Initialize a variable 'test' to 0  Scanner in = new Scanner(System.in); // Create a Scanner object for user input  System.out.print("Input a positive integer: "); // Prompt the user to input a positive integer    int n = in.nextInt(); // Read the user's input as an integer  if (n < 1) {  System.out.print(Boolean.toString(false)); // If n is less than 1, print "false" and set 'test' to 1  test = 1;  }  if ((n & (n - 1)) != 0) {  System.out.print(Boolean.toString(false)); // If n is not a power of 2, print "false" and set 'test' to 1  test = 1;  }  if (test == 0) {  System.out.print(Boolean.toString((n & 0x55555555) != 0)); // If 'test' is 0, check if n has odd bits set and print the result  }  System.out.print("\n"); // Print a new line  }  } |

Output:



**111.** Write a Java program to add two numbers without arithmetic operators.  
Given x = 10 and y = 12; result = 22  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.Scanner;  public class Example111 {  public static void main(String[] arg) {  int x, y; // Declare two integer variables, 'x' and 'y'  Scanner in = new Scanner(System.in); // Create a Scanner object for user input  System.out.print("Input first number: "); // Prompt the user to input the first number  x = in.nextInt(); // Read and store the first number from the user  System.out.print("Input second number: "); // Prompt the user to input the second number  y = in.nextInt(); // Read and store the second number from the user  while (y != 0) {  int carry = x & y; // Calculate the carry by bitwise AND operation between x and y  x = x ^ y; // Calculate the sum without considering the carry by bitwise XOR operation  y = carry << 1; // Calculate the carry for the next iteration by shifting it left by one position  }  System.out.print("Sum: " + x); // Print the sum of the two numbers  System.out.print("\n"); // Print a new line  }  } |

Output:

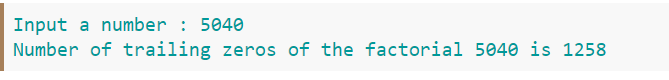
A number on a white background

Description automatically generated

**112.** Write a Java program to compute the number of trailing zeros in a factorial.  
7! = 5040, therefore the output should be 1  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.Scanner;  public class Example112 {  public static void main(String[] arg) {  Scanner in = new Scanner(System.in); // Create a Scanner object for user input  System.out.print("Input a number: "); // Prompt the user to input a number  int n = in.nextInt(); // Read and store the user's input as 'n'  int n1 = n; // Create a copy of 'n' to preserve the original value  long ctr = 0; // Initialize a variable 'ctr' to count trailing zeros  while (n != 0) {  ctr += n / 5; // Count the number of trailing zeros by dividing 'n' by 5 and accumulating the result  n /= 5; // Reduce 'n' by dividing it by 5 for the next iteration  }  System.out.printf("Number of trailing zeros of the factorial %d is %d ", n1, ctr); // Print the result  System.out.printf("\n"); // Print a new line  }  } |

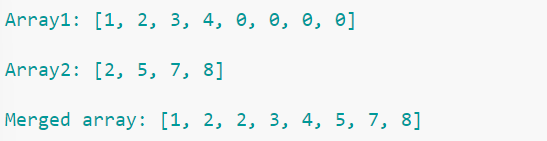
Output:



**113.** Write a Java program to merge two given sorted arrays of integers and create another sorted array.  
array1 = [1,2,3,4]  
array2 = [2,5,7, 8]  
result = [1,2,2,3,4,5,7,8]  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  public class Example113 {  public static void main(String[] arg) {  // Declare two sorted integer arrays, array1 and array2  // array1 has 'm' elements but is large enough to accommodate 'm+n' elements  // array2 has 'n' elements    // Declaration and instantiation of array1  int array1[] = new int[8];    // Initialize the first four elements of array1  array1[0] = 1;  array1[1] = 2;  array1[2] = 3;  array1[3] = 4;    // Initialize array2  int[] array2 = {2, 5, 7, 8};    System.out.println("\nArray1: " + Arrays.toString(array1));  System.out.println("\nArray2: " + Arrays.toString(array2));    // Define variables m and n  int m = 4, n = 4;    // Initialize pointers for array1 (i) and array2 (j) and the index for merging (index)  int i = m - 1, j = n - 1, index = m + n - 1;    // Merge the two arrays into array1  while (i >= 0 && j >= 0) {  if (array1[i] > array2[j]) {  array1[index--] = array1[i--];  } else {  array1[index--] = array2[j--];  }  }    // Handle remaining elements in array1 (if any)  while (i >= 0) {  array1[index--] = array1[i--];  }    // Handle remaining elements in array2 (if any)  while (j >= 0) {  array1[index--] = array2[j--];  }  System.out.println("\nMerged array: " + Arrays.toString(array1));  }  } |

Output:



**114.** Write a Java program that rotates a string by an offset (rotate from left to right.  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

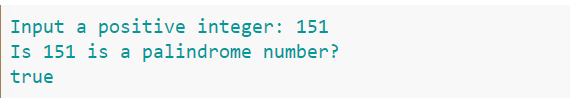
|  |
| --- |
| import java.util.\*;  public class Example114 {  public static void main(String[] arg) {  // Input string  String str = "abcdef";    // Convert the string to a character array  char[] A = str.toCharArray();    // Define the offset for rotation  int offset = 3;    // Calculate the length of the character array  int len = A.length;    // Ensure that the offset is within the bounds of the array  offset %= len;    // Reverse the first portion of the array  reverse(A, 0, len - offset - 1);    // Reverse the second portion of the array  reverse(A, len - offset, len - 1);    // Reverse the entire array to complete the rotation  reverse(A, 0, len - 1);    // Print the rotated array  System.out.println("\n" + Arrays.toString(A));  }  // Helper function to reverse a portion of a character array  private static void reverse(char[] str, int start, int end) {  while (start < end) {  char temp = str[start];  str[start] = str[end];  str[end] = temp;  start++;  end--;  }  }  } |

Output:

**115.** Write a Java program to check if a positive number is a palindrome or not.  
Input a positive integer: 151  
Is 151 is a palindrome number?  
true  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  public class test {  public static void main(String[] args) {  int num;    // Create a Scanner object for user input  Scanner in = new Scanner(System.in);    // Prompt the user for a positive integer  System.out.print("Input a positive integer: ");    // Read the integer entered by the user  int n = in.nextInt();    // Display a message to check if the number is a palindrome  System.out.printf("Is %d a palindrome number?\n", n);    // Check if the number is a palindrome and print the result  System.out.println(is\_Palindrome(n));  }  // Function to reverse the digits of a number  public static int reverse\_nums(int n) {  int reverse = 0;  while (n != 0) {  reverse \*= 10;  reverse += n % 10;  n /= 10;  }  return reverse;  }  // Function to check if a number is a palindrome  public static boolean is\_Palindrome(int n) {  return (n == reverse\_nums(n));  }  } |

Output:

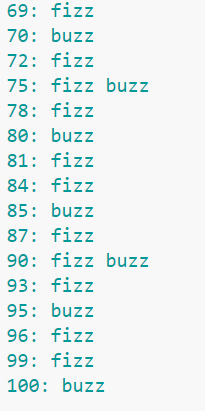
**116.** Write a Java program that iterates integers from 1 to 100. For multiples of three print "Fizz" instead of the number and print "Buzz" for five. When the number is divided by three and five, print "fizz buzz".  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  public class Exercise116 {  public static void main(String[] args) {  // Iterate from 1 to 100  for (int i = 1; i <= 100; i++) {  if (i % 3 == 0 && i % 5 == 0) {  // Check if the number is divisible by both 3 and 5 (fizz buzz)  System.out.printf("\n%d: fizz buzz", i);  } else if (i % 5 == 0) {  // Check if the number is divisible by 5 (buzz)  System.out.printf("\n%d: buzz", i);  } else if (i % 3 == 0) {  // Check if the number is divisible by 3 (fizz)  System.out.printf("\n%d: fizz", i);  }  }  System.out.printf("\n");  }  } |

Output:

A screenshot of a computer screen

Description automatically generated



**117.** Write a Java program to compute the square root of a given number.  
Input a positive integer: 25  
Square root of 25 is: 5  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  public class Exercise117 {  public static void main(String[] args) {  int num;  Scanner in = new Scanner(System.in);  // Prompt the user to input a positive integer  System.out.print("Input a positive integer: ");  int n = in.nextInt();  // Print a message indicating the calculation about to take place  System.out.printf("Square root of %d is: ", n);  // Call the sqrt method to calculate the square root and print the result  System.out.println(sqrt(n));  }  // Method to calculate the square root of a number  private static int sqrt(int num) {  if (num == 0 || num == 1) {  return num;  }  int a = 0;  int b = num;  // Perform a binary search to find the square root  while (a <= b) {  int mid = (a + b) >> 1;  if (num / mid < mid) {  b = mid - 1;  } else {  if (num / (mid + 1) <= mid) {  return mid;  }  a = mid + 1;  }  }  return a;  }  } |

Output:

A close up of numbers

Description automatically generated**118.** Write a Java program to get the first occurrence (Position starts from 0.) of a string within a given string.  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.Scanner;  public class Main {  public static void main(String[] args) {  // Define two strings, 's' and 't'  String s = "Python";  //String t = "Py";  String t = "yt";  // String t = "ab";  // Call the 'strStr' function with 's' and 't' as arguments and print the result  System.out.printf(String.valueOf(strStr(s, t)));  }  public static int strStr(String source, String target) {  // Check if either 'source' or 'target' is null  if (source == null || target == null) {  return -1; // Return -1 if either of them is null  }    // Check if 'target' is an empty string or is equal to 'source'  if ("".equals(target) || source.equals(target)) {  return 0; // Return 0 if 'target' is empty or equal to 'source'  }    int i = 0;  int last = source.length() - target.length() + 1;  // Iterate through 'source' while there are characters left to compare  while (i < last) {  if (source.charAt(i) == target.charAt(0)) {  boolean equal = true;  // Check if characters in 'source' match 'target' from the current position  for (int j = 0; j < target.length() && equal; ++j) {  if (source.charAt(i + j) != target.charAt(j)) {  equal = false;  }  }  // If 'target' matches a substring of 'source', return the starting index  if (equal) {  return i;  }  }  ++i;  }  // If 'target' is not found in 'source', return -1  return -1;  }  } |

Output:

A white rectangular object with a black border

Description automatically generated

**119.** Write a Java program to get the first occurrence (Position starts from 0.) of an element of a given array.  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.Scanner;  public class Main {  public static void main(String[] args) {  // Declare an array of integers 'nums' and an integer 'target'  int nums[] = {2, 4, 6, 7, 8};  int target = 7;    // Initialize 'lower' to the start of the array and 'upper' to the end of the array  int lower = 0;  int upper = nums.length - 1;    // Initialize 'index' to -1; it will store the position of the 'target'  int index = -1;    // Perform a binary search to find the 'target' in the 'nums' array  while (lower <= upper) {  // Calculate the middle index 'mid'  int mid = (lower + upper) >> 1;    // Check if 'nums[mid]' is equal to the 'target'  if (nums[mid] == target) {  index = mid; // Set 'index' to the position of the 'target'  }    // Adjust 'lower' and 'upper' based on the comparison with 'target'  if (nums[mid] >= target) {  upper = mid - 1; // Move 'upper' to the left half  } else {  lower = mid + 1; // Move 'lower' to the right half  }  }    // Print the position of the 'target' in the array  System.out.print("Position of " + target + " is " + index);  }  } |

Output:



**120.** Write a Java program that searches for a value in an m x n matrix.  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.Scanner;  public class Main {  public static void main(String[] args) {  // Initialize the 'target' value for searching  int target = 0;    // Create a 3x3 matrix and populate it with values  int[][] matrix = new int[3][3];  for (int row = 0; row < 3; row++) {  for (int col = 0; col < 3; col++) {  matrix[row][col] = (1 + row \* 3 + col);  }  }    // Display the contents of the matrix  for (int row = 0; row < 3; row++) {  for (int col = 0; col < 3; col++) {  System.out.print(matrix[row][col] + " ");  if (col == 2) {  System.out.println();  }  }  }    // Print the result of the 'searchMatrix' function  System.out.print(Boolean.toString(searchMatrix(matrix, target)));  }    public static boolean searchMatrix(int[][] matrix, int target) {  // Check if the matrix is empty or has no columns  if (matrix.length == 0 || matrix[0].length == 0) {  return false;  }    // Get the number of rows 'm' and columns 'n' in the matrix  int m = matrix.length;  int n = matrix[0].length;    // Initialize 'lower' and 'higher' for binary search  int lower = 0;  int higher = m \* n - 1;    // Perform binary search to find 'target' in the matrix  while (lower <= higher) {  int mid = (lower + higher) >> 1;  int val = matrix[mid / n][mid % n];  if (val == target) {  return true;  }  if (val < target) {  lower = mid + 1;  } else {  higher = mid - 1;  }  }    // If 'target' is not found, return false  return false;  }  } |

Output:

A white background with black and white clouds

Description automatically generated with medium confidence

**121.** Write a Java program to reverse a linked list.  
Example: For linked list 20->40->60->80, the reversed linked list is 80->60->40->20  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| class LinkedList {  // Static variable to store the head of the linked list  static Node head;  static class Node {  int data; // Data stored in the node  Node next\_node; // Reference to the next node  Node(int d) {  data = d;  next\_node = null;  }  }  // Function to reverse the linked list  Node reverse(Node node) {  Node prev\_node = null;  Node current\_node = node;  Node next\_node = null;  while (current\_node != null) {  next\_node = current\_node.next\_node;  current\_node.next\_node = prev\_node;  prev\_node = current\_node;  current\_node = next\_node;  }  node = prev\_node;  return node;  }  // Function to print the elements of the linked list  void printList(Node node) {  while (node != null) {  System.out.print(node.data + " ");  node = node.next\_node;  }  }  public static void main(String[] args) {  LinkedList list = new LinkedList();    // Create a linked list with some initial values  list.head = new Node(20);  list.head.next\_node = new Node(40);  list.head.next\_node.next\_node = new Node(60);  list.head.next\_node.next\_node.next\_node = new Node(80);    // Print the original linked list  System.out.println("Original Linked list:");  list.printList(head);    // Reverse the linked list  head = list.reverse(head);    System.out.println("");  // Print the reversed linked list  System.out.println("Reversed Linked list:");  list.printList(head);  }  } |

Output:

A white background with blue text

Description automatically generated

**122.** Write a Java program to find a contiguous subarray with the largest sum from a given array of integers.  
Note: In [computer](https://www.w3resource.com/java-exercises/basic/index.php) science, the maximum subarray problem is the task of finding the contiguous subarray within a one-dimensional array of numbers which has the largest sum. For example, for the sequence of values −2, 1, −3, 4, −1, 2, 1, −5, 4; the contiguous subarray with the largest sum is 4, −1, 2, 1, with sum 6. The subarray should contain one integer at least.

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.Scanner;  public class Main {  public static void main(String[] args) {  // Input array  int[] nums = {-2, 1, -3, 4, -1, 2, 1, -5, 4};  // Find and print the maximum subarray sum  System.out.print(max\_SubArray(nums));  }  public static int max\_SubArray(int[] nums) {  // Check if the input array is empty  if (nums.length < 1) {  return 0;  }  // Initialize variables to track the maximum sum, its start and end indices  int max = nums[0];  int max\_Begin = 0;  int max\_End = 0;  int begin = 0;  int end = 0;  int sum = 0;    while (end < nums.length) {  // Update the current sum with the value at the current end index  sum += nums[end];  if (sum < 0) {  // If the current sum becomes negative, reset it and update the beginning index  sum = 0;  begin = end + 1;  } else {  // If the current sum is greater than the maximum, update the maximum  if (sum > max) {  max = sum;  max\_Begin = begin;  max\_End = end;  }  }  end++;  }  // Return the maximum sum of the subarray  return max;  }  } |

Output:



**123.** Write a Java program to find the subarray with smallest sum from a given array of integers.  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  public class Main {  public static void main(String[] args) {  // Create an ArrayList to store integers  ArrayList nums = new ArrayList();  nums.add(-2);  nums.add(1);  nums.add(-3);  nums.add(4);  // Call the min\_SubArray function and print the result  System.out.print(min\_SubArray(nums));  }  public static int min\_SubArray(ArrayList nums) {  // Create an array to store the same integers for dynamic programming  int[] nums1 = new int[nums.size()];  nums1[0] = nums.get(0);  // Initialize the minimum value to the first element  int min = nums1[0];  // Loop through the ArrayList to calculate minimum subarray sum  for (int i = 1; i < nums.size(); ++i) {  // Calculate the minimum of the current element and the sum of the previous subarray  nums1[i] = Math.min(nums.get(i), nums.get(i) + nums1[i - 1]);  // Update the minimum value if needed  min = Math.min(min, nums1[i]);  }  // Return the minimum subarray sum  return min;  }  } |

Output:

A white background with a black border

Description automatically generated with medium confidence

**124.** Write a Java program to find the index of a value in a sorted array. If the value does not find return the index where it would be if it were inserted in order.  
Example:  
[1, 2, 4, 5, 6] 5(target) -> 3(index)  
[1, 2, 4, 5, 6] 0(target) -> 0(index)  
[1, 2, 4, 5, 6] 7(target) -> 5(index)  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  public class Main {  public static void main(String[] args) {  // Create an array of integers  int[] nums = {1, 2, 4, 5, 6};  int target = 5;  // target = 0;  // target = 7;  // Call the searchInsert function and print the result  System.out.print(searchInsert(nums, target));  }  public static int searchInsert(int[] nums1, int target) {  // Check if the input array is empty or null  if (nums1 == null || nums1.length == 0) {  return 0;  }  // Initialize variables for binary search  int start = 0;  int end = nums1.length - 1;  int mid = start + (end - start) / 2;  while (start + 1 < end) {  mid = start + (end - start) / 2;  // Compare the middle element with the target  if (nums1[mid] == target) {  return mid;  } else if (nums1[mid] > target) {  end = mid;  } else {  start = mid;  }  }    // Determine the insertion position based on binary search results  if (nums1[start] >= target) {  return start;  } else if (nums1[start] < target && target <= nums1[end]) {  return end;  } else {  return end + 1;  }  }  } |

Output:

**125.** Write a Java program to get the preorder traversal of the values of the nodes in a binary tree.  
Example:

10

/ \

20 30

/ \

40 50

Expected output: 10 20 40 50 30  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| class Node  {  int key;  Node left, right;  public Node(int item)  {  // Constructor to create a new Node with the given item  key = item;  left = right = null;  }  }  class BinaryTree  {  Node root;  BinaryTree()  {  // Constructor to create an empty binary tree  root = null;  }  void print\_Preorder(Node node)  {  if (node == null)  return;  // Print the key of the current node  System.out.print(node.key + " ");  // Recursively print the left subtree in preorder  print\_Preorder(node.left);  // Recursively print the right subtree in preorder  print\_Preorder(node.right);  }  void print\_Preorder()  {  // Wrapper method to start printing the tree in preorder  print\_Preorder(root);  }  public static void main(String[] args)  {  BinaryTree tree = new BinaryTree();    // Create a binary tree with nodes and keys  tree.root = new Node(55);  tree.root.left = new Node(21);  tree.root.right = new Node(80);  tree.root.left.left = new Node(9);  tree.root.left.right = new Node(29);  tree.root.right.left = new Node(76);  tree.root.right.right = new Node(91);  // Display a message and initiate the preorder traversal  System.out.println("Preorder traversal of binary tree is: ");  tree.print\_Preorder();  }  } |

Output:

A close-up of a word

Description automatically generated**126.** Write a Java program to get the in-order traversal of its nodes' values in a binary tree.

10

/ \

20 30

/ \

40 50

Example:{10, 20, 30, 40, 50}  
Output: 40 20 50 10 30  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| class Node  {  int key;  Node left, right;  public Node(int item)  {  // Constructor to create a new Node with the given item  key = item;  left = right = null;  }  }  class BinaryTree  {  Node root;  BinaryTree()  {  // Constructor to create an empty binary tree  root = null;  }  void print\_Inorder(Node node)  {  if (node == null)  return;  // Recursively print the left subtree in inorder  print\_Inorder(node.left);  // Print the key of the current node  System.out.print(node.key + " ");  // Recursively print the right subtree in inorder  print\_Inorder(node.right);  }  void print\_Inorder()  {  // Wrapper method to start printing the tree in inorder  print\_Inorder(root);  }  public static void main(String[] args)  {  BinaryTree tree = new BinaryTree();  // Create a binary tree with nodes and keys  tree.root = new Node(55);  tree.root.left = new Node(21);  tree.root.right = new Node(80);  tree.root.left.left = new Node(9);  tree.root.left.right = new Node(29);  tree.root.right.left = new Node(76);  tree.root.right.right = new Node(91);  // Display a message and initiate the inorder traversal  System.out.println("\nInorder traversal of binary tree is: ");  tree.print\_Inorder();  }  } |

Output:

A close-up of a word

Description automatically generated

**127.** Write a Java program to get the Postorder traversal of its nodes' values in a binary tree.

10

/ \

20 30

/ \

40 50

[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| class Node  {  int key;  Node left, right;  public Node(int item)  {  // Constructor to create a new Node with the given item  key = item;  left = right = null;  }  }  class BinaryTree  {  // Root of Binary Tree  Node root;  BinaryTree()  {  // Constructor to create an empty binary tree  root = null;  }  void print\_Postorder(Node node)  {  if (node == null)  return;  // Recursively print the left subtree in postorder  print\_Postorder(node.left);  // Recursively print the right subtree in postorder  print\_Postorder(node.right);  // Print the key of the current node  System.out.print(node.key + " ");  }  void print\_Postorder()  {  // Wrapper method to start printing the tree in postorder  print\_Postorder(root);  }  public static void main(String[] args)  {  BinaryTree tree = new BinaryTree();  // Create a binary tree with nodes and keys  tree.root = new Node(55);  tree.root.left = new Node(21);  tree.root.right = new Node(80);  tree.root.left.left = new Node(9);  tree.root.left.right = new Node(29);  tree.root.right.left = new Node(76);  tree.root.right.right = new Node(91);  // Display a message and initiate the postorder traversal  System.out.println("\nPostorder traversal of binary tree is: ");  tree.print\_Postorder();  }  } |

Output:

A close-up of a word

Description automatically generated**128.** Write a Java program to calculate the median of a non-sorted array of integers.  
Original array: [10, 2, 38, 22, 38, 23]  
Median of the said array of integers: 30  
Original array: [10, 2, 38, 23, 38, 23, 21]  
Median of the said array of integers: 23  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  public class Main {  public static void main(String[] args) {  // Define and initialize an array of integers  int[] nums = {10, 2, 38, 22, 38, 23};    // Display the original array  System.out.println("Original array: " + Arrays.toString(nums));    // Calculate and display the median of the array  System.out.println("Median of the said array of integers: " + getMedian(nums));    // Define and initialize another array of integers  int[] nums1 = {10, 2, 38, 23, 38, 23, 21};    // Display the original array  System.out.println("\nOriginal array: " + Arrays.toString(nums1));    // Calculate and display the median of the second array  System.out.println("Median of the said array of integers: " + getMedian(nums1));  }  public static int getMedian(int[] array) {  // Check if the length of the array is even  if (array.length % 2 == 0) {  // Calculate the median for even-sized arrays  int mid = array.length / 2;  return (array[mid] + array[mid - 1]) / 2;  }  // Calculate the median for odd-sized arrays  return array[array.length / 2];  }  } |

Output:

A white background with blue text

Description automatically generated**129.** Write a Java program to find a number that appears only once in a given array of integers. All numbers occur twice.  
Source Array : [10, 20, 10, 20, 30, 40, 40, 30, 50] 50 appears only once  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  public class Main {  public static void main(String[] args) {  // Define an array of integers  int nums[] = {10, 20, 10, 20, 30, 40, 40, 30, 50};    // Declare a variable to store the result  int result;    // Display the source array  System.out.println("Source Array : " + Arrays.toString(nums));    // Calculate and display the number that appears only once  result = getSingleNumber(nums);  System.out.println("\n" + result + " appears only once.");  }  public static int getSingleNumber(int[] nums) {  // Check if the array is null or empty  if (nums == null || nums.length == 0) {  return -1;  }    // Initialize the result to 0  int result = 0;    // Calculate the number that appears only once using XOR (^) operator  for (int i = 0; i < nums.length; i++) {  result ^= nums[i];  }    // Return the result  return result;  }  } |

Output:

A number on a white background

Description automatically generated**130.** Write a Java program to find the maximum depth of a given binary tree.  
Sample Output: The Maximum depth of the binary tree is: 3  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| class Node {  int data;  Node left, right;  public Node(int item) {  // Constructor to initialize a node with given data  data = item;  left = right = null;  }  }  public class BinaryTree {  Node root;  public int maxDepth(Node root) {  // Recursive function to calculate the maximum depth of the binary tree  if (root == null)  return 0;  int left\_Depth = maxDepth(root.left);  int right\_Depth = maxDepth(root.right);  int bigger = Math.max(left\_Depth, right\_Depth);  return bigger + 1;  }  public static void main(String args[]) {  BinaryTree tree = new BinaryTree();  tree.root = new Node(55);  tree.root.left = new Node(21);  tree.root.right = new Node(80);  tree.root.left.left = new Node(9);  tree.root.left.right = new Node(29);  tree.root.right.left = new Node(76);  tree.root.right.right = new Node(91);  // Create a binary tree and calculate its maximum depth  System.out.println("The Maximum depth of the binary tree is: " + tree.maxDepth(tree.root));  }  } |

Output:

A close up of a word

Description automatically generated**131.** Write a Java program to find the updated length of a sorted array where each element appears only once (remove duplicates).  
Original array: [1, 1, 2, 3, 3, 3, 4, 5, 6, 7, 7] The length of the original array is: 11 After removing duplicates, the new length of the array is: 7  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| class Node {  int data;  Node left, right;  public Node(int item) {  // Constructor to initialize a node with given data  data = item;  left = right = null;  }  }  public class BinaryTree {  Node root;  public int maxDepth(Node root) {  // Recursive function to calculate the maximum depth of the binary tree  if (root == null)  return 0;  int left\_Depth = maxDepth(root.left);  int right\_Depth = maxDepth(root.right);  int bigger = Math.max(left\_Depth, right\_Depth);  return bigger + 1;  }  public static void main(String args[]) {  BinaryTree tree = new BinaryTree();  tree.root = new Node(55);  tree.root.left = new Node(21);  tree.root.right = new Node(80);  tree.root.left.left = new Node(9);  tree.root.left.right = new Node(29);  tree.root.right.left = new Node(76);  tree.root.right.right = new Node(91);  // Create a binary tree and calculate its maximum depth  System.out.println("The Maximum depth of the binary tree is: " + tree.maxDepth(tree.root));  }  } |

Output:



**132.** Write a Java program to find the updated length of a given sorted array where duplicate elements appear at most twice.  
Original array: [1, 1, 2, 3, 3, 3, 4, 5, 6, 7, 7, 7, 7]  
The length of the original array is: 13  
After removing duplicates, the new length of the array is: 10  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.Arrays;  class Solution {  // Static method to remove duplicates from the given array, allowing at most two duplicates  static int remove\_Duplicates\_twice(int[] nums) {  // Check for invalid or empty input array  if (nums == null || nums.length == 0) {  return 0;  }  int index = 1; // Initialize the index for the resulting array  for (int i = 2; i < nums.length; i++) {  // Check if the current element is different from the element at 'index',  // or if it's the second occurrence of a duplicate, but not the third  if (nums[i] != nums[index] || (nums[i] == nums[index] && nums[i] != nums[index - 1])) {  index++; // Increment the index for the resulting array  nums[index] = nums[i]; // Copy the unique or second occurrence of a duplicate element  }  }  // The new length of the array is one more than the 'index'  return index + 1;  }  /\* Driver program to test above functions \*/  public static void main(String[] args) {  int[] nums = {1, 1, 2, 3, 3, 3, 4, 5, 6, 7, 7, 7, 7};  System.out.println("Original array: " + Arrays.toString(nums));  System.out.println("The length of the original array is: " + nums.length);  System.out.println("After removing duplicates, the new length of the array is: " + remove\_Duplicates\_twice(nums));  }  } |

Output:

A close up of blue text

Description automatically generated**133.** Write a Java program to find a path from top left to bottom in the right direction which minimizes the sum of all numbers along its path.  
Note: Move either down or right at any point in time.  
Sample Output: Sum of all numbers along its path: 13  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*  public class Solution {  // Static method to find the minimum path sum in a 2D grid  public static int minPathSum(int[][] grid) {  // Check for invalid or empty input grid  if (grid == null || grid.length == 0 || grid[0] == null || grid[0].length == 0) {  return 0;  }  int m = grid.length; // Number of rows in the grid  int n = grid[0].length; // Number of columns in the grid  int[][] temp = new int[m][n]; // Temporary array to store minimum path sum  for (int i = 0; i < m; i++) {  for (int j = 0; j < n; j++) {  if (i == 0 && j == 0) {  temp[i][j] = grid[i][j]; // Initialize the starting point  continue;  }  // Calculate the minimum path sum from either the cell above or the cell on the left  int from\_up = i == 0 ? Integer.MAX\_VALUE : temp[i - 1][j];  int from\_left = j == 0 ? Integer.MAX\_VALUE : temp[i][j - 1];  temp[i][j] = Math.min(from\_up, from\_left) + grid[i][j]; // Update the temporary array  }  }  // Return the minimum path sum for the last cell  return temp[m - 1][n - 1];  }  public static void main(String[] args) {  // Example grid  int[][] grid = new int[][] {{7, 4, 2},  {0, 5, 6},  {3, 1, 2}};    System.out.println("Sum of all numbers along its path: " + minPathSum(grid));  }  } |

Output:

A close up of a text

Description automatically generated**134.** Write a Java program to find distinct ways to climb to the top (n steps to reach the top) of stairs. Each time you climb, you can climb 1 or 2 steps.  
Example: n = 5  
a) 1+1+1+1+1 = 5 b) 1+1+1+2 = 5 c) 1+2+2 = 5 d) 2+2+1 = 5 e) 2+1+1+1 = 5 f) 2+1+2 = 5 g) 1+2+1+1 = 5 h) 1+1+2+1 = 5  
Sample Output: Distinct ways can you climb to the top: 8  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*  class Solution {  // Static method to calculate the distinct ways to climb stairs  public static int climbStairs(int n) {  if (n <= 1) {  return 1; // If there is 0 or 1 step, there is only 1 way to climb.  }  int[] s\_case = new int[n + 1]; // Create an array to store the number of distinct ways for each step count.  s\_case[0] = 1; // There is 1 way to climb 0 steps.  s\_case[1] = 1; // There is 1 way to climb 1 step.  for (int i = 2; i <= n; i++) {  // Calculate the number of distinct ways for each step by adding the ways from the previous two steps.  s\_case[i] = s\_case[i - 1] + s\_case[i - 2];  }  return s\_case[n]; // Return the number of distinct ways to climb n steps.  }  public static void main(String[] args) {  int steps = 5; // The number of steps to climb  System.out.println("Distinct ways can you climb to the top: " + climbStairs(steps));  }  } |

Output:

A close up of a word

Description automatically generated**135.** Write a Java program to remove duplicates from a sorted linked list.  
Original List with duplicate elements:  
12->12->13->14->15->15->16->17->17  
After removing duplicates from the said list:  
12->13->14->15->16->17  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| class LinkedList {  Node head; // Reference to the head node of the linked list  /\* Linked list Node \*/  class Node {  int data; // Data stored in the node  Node next; // Reference to the next node  Node(int d) {  data = d;  next = null;  }  }  // Method to remove duplicate elements from the linked list  void remove\_Duplicates() {  Node current = head; // Initialize the current node to the head  Node next\_next; // Reference to the next of the next node  if (head == null)  return; // If the linked list is empty, return  while (current.next != null) {  if (current.data == current.next.data) {  next\_next = current.next.next;  current.next = null; // Remove the duplicate node  current.next = next\_next; // Update the next reference  } else  current = current.next; // Move to the next node  }  }  // Method to insert a new node at the front of the linked list  public void push(int new\_data) {  Node new\_node = new Node(new\_data); // Create a new node  new\_node.next = head; // Set the next of the new node to the current head  head = new\_node; // Update the head to the new node  }  // Method to print the linked list  void printList() {  Node temp = head; // Initialize a temporary node with the head  while (temp != null) {  System.out.print(temp.data); // Print the data of the current node  if (temp.next != null) {  System.out.print("->"); // Print an arrow to indicate the next node  }  temp = temp.next; // Move to the next node  }  System.out.println(); // Print a new line to complete the list  }  // Driver program to test the above functions  public static void main(String args[]) {  LinkedList l\_list = new LinkedList();    // Insert data into the linked list  l\_list.push(17);  l\_list.push(17);  l\_list.push(16);  l\_list.push(15);  l\_list.push(15);  l\_list.push(14);  l\_list.push(13);  l\_list.push(12);  l\_list.push(12);    System.out.println("Original List with duplicate elements:");  l\_list.printList();    l\_list.remove\_Duplicates(); // Remove duplicates    System.out.println("After removing duplicates from the said list:");  l\_list.printList();  }  } |

Output:

A white background with blue text

Description automatically generated**136.** Write a Java program to find possible distinct paths from the top-left corner to the bottom-right corner of a given grid (m x n).  
Note: You can move either down or right at any point in time.  
Sample Output: Unique paths from top-left corner to bottom-right corner of the said grid: 3  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*  public class Solution {  /\*\*  \* @param n, m: positive integer (1 <= n ,m <= 100)  \* @return an integer  \*/  public static int unique\_Paths(int m, int n) {  if (m <= 0 || n <= 0) {  return 0;  }  int[][] grid = new int[m][n]; // Create a grid to store the number of unique paths  // Nested loops to iterate through the grid  for (int i = m - 1; i >= 0; --i) { // Start from the bottom row and move upwards  for (int j = n - 1; j >= 0; --j) { // Start from the rightmost column and move leftwards  grid[i][j] = get\_Paths(grid, i, j); // Calculate the unique paths for the current cell  }  }  return grid[0][0]; // The top-left corner now contains the total unique paths  }    // Helper function to calculate unique paths for a cell  private static int get\_Paths(int[][] grid, int i, int j) {  if (i >= grid.length - 1 || j >= grid[0].length - 1) {  return 1; // If at the rightmost column or bottom row, there is only one path  }  return grid[i][j + 1] + grid[i + 1][j]; // Sum the unique paths from the right and bottom cells  }    // Main method for testing the unique\_Paths function  public static void main(String[] args) {  int m = 3;  int n = 2;  System.out.println("Unique paths from top-left corner to bottom-right corner of the said grid: "+unique\_Paths(m, n));  }  } |

Output:



**137.** Write a Java program to find possible unique paths considering some obstacles, from top-left corner to bottom-right corner of a given grid (m x n).  
Note: You can move either down or right at any point in time and an obstacle and empty space is marked as 1 and 0 respectively in the grid.  
Sample grid:  
int[][] obstacle\_Grid ={  
{0, 0, 0},  
{0, 1, 0},  
{0, 0, 0},  
};  
Sample Output: Unique paths from top-left corner to bottom-right corner of the said grid (considering some obstacles): 2  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*  public class Solution {  public static int uniquePaths\_With\_obstacle\_Grid(int[][] obstacle\_Grid) {  // Get the number of rows (m) in the obstacle grid  int m = obstacle\_Grid.length;  if (m <= 0) {  return 0;  }  // Get the number of columns (n) in the obstacle grid  int n = obstacle\_Grid[0].length;  if (n <= 0) {  return 0;  }  // Create a dynamic programming array to store unique paths  int[][] dp = new int[m + 1][n + 1];  dp[m][n - 1] = 1;  // Nested loops to iterate through the grid  for (int i = m - 1; i >= 0; --i) { // Start from the bottom row and move upwards  for (int j = n - 1; j >= 0; --j) { // Start from the rightmost column and move leftwards  // Calculate unique paths for the current cell based on obstacles  dp[i][j] = (obstacle\_Grid[i][j] == 0) ? dp[i + 1][j] + dp[i][j + 1] : 0;  }  }  // Return the count of unique paths from top-left to bottom-right corner  return dp[0][0];  }    // Main method for testing the uniquePaths\_With\_obstacle\_Grid function  public static void main(String[] args) {  int[][] obstacle\_Grid ={  {0, 0, 0},  {0, 1, 0},  {0, 0, 0},  };  System.out.println("Unique paths from top-left corner to bottom-right corner of the said grid (considering some obstacles): "+uniquePaths\_With\_obstacle\_Grid(obstacle\_Grid));  }  } |

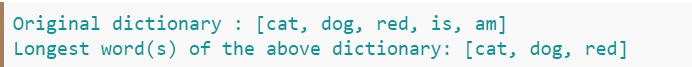
Output:



**138.** Write a Java program to find the longest words in a dictionary.  
Example-1:  
{  
"cat",  
"flag",  
"green",  
"country",  
"w3resource"  
}  
Result: "w3resource"  
Example-2:  
{  
"cat",  
"dog",  
"red",  
"is",  
"am"  
}  
Result: "cat", "dog", "red"  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  public class Solution {  // Function to find and return the longest words in the given dictionary  static ArrayList longestWords(String[] dictionary) {  ArrayList list = new ArrayList();  int longest\_length = 0;  // Iterate through each word in the dictionary  for (String str : dictionary) {  int length = str.length();    // Check if the current word is longer than the previously found longest word(s)  if (length > longest\_length) {  longest\_length = length;  list.clear(); // Clear the list as a new longest word is found  }    // If the current word has the same length as the longest word(s), add it to the list  if (length == longest\_length) {  list.add(str);  }  }  return list; // Return the list of longest words  }    public static void main(String[] args) {  // Sample dictionary containing words  // String[] dict = {"cat", "flag", "green", "country", "w3resource"};  String[] dict = {"cat", "dog", "red", "is", "am"};  // Print the original dictionary and the longest word(s)  System.out.println("Original dictionary: " + Arrays.toString(dict));  System.out.println("Longest word(s) of the above dictionary: " + longestWords(dict));  }  } |

Output:



**139.** Write a Java program to get the index of the first and the last number of a subarray where the sum of numbers is zero. This is from a given array of integers.  
Original Array : [1, 2, 3, -6, 5, 4]  
Index of the subarray of the said array where the sum of numbers is zero: [0, 3]  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  public class Solution {  public static List<Integer> subarraySum(int[] nums) {  List<Integer> temp = new ArrayList<>();    // Check if the input array is null or empty, and return an empty list if so  if (nums == null || nums.length == 0) {  return temp;  }    int pre\_Sum = 0;  Map<Integer, Integer> map = new HashMap<>();  map.put(pre\_Sum, -1);    // Iterate through the elements of the input array  for (int i = 0; i < nums.length; i++) {  pre\_Sum += nums[i];    // Check if the current prefix sum already exists in the map  if (map.containsKey(pre\_Sum)) {  temp.add(map.get(pre\_Sum) + 1); // Add the start index of the subarray  temp.add(i); // Add the end index of the subarray  return temp; // Return the list of indices  }  // If the prefix sum is not found, add it to the map with its index  map.put(pre\_Sum, i);  }    return temp; // Return an empty list if no subarray with a sum of zero is found  }    public static void main(String[] args) {  int[] nums = {1, 2, 3, -6, 5, 4};  // Print the original array and the indices of the subarray with a sum of zero  System.out.println("Original Array: " + Arrays.toString(nums));  System.out.println("Index of the subarray of the said array where the sum of numbers is zero: " + subarraySum(nums));  }  } |

Output:

**140.** Write a Java program to merge all overlapping intervals from a given collection of intervals.  
Sample Output: 1 6  
8 10  
15 20  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  public class Solution  {  public static void main (String[] args) throws java.lang.Exception  {  // Create an ArrayList of Interval objects  ArrayList<Interval> x = new ArrayList<>();  // Add intervals to the ArrayList  x.add(new Interval(1, 3));  x.add(new Interval(2, 6));  x.add(new Interval(8, 10));  x.add(new Interval(15, 18));  x.add(new Interval(17, 20));  // Merge overlapping intervals  x = merge(x);  // Print the merged intervals  for(Interval i : x)  {  System.out.println(i.getStart() + " " + i.getEnd());  }  }  public static ArrayList<Interval> merge(ArrayList<Interval> intervals) {  // Check for the number of intervals  if(intervals.size() == 0 || intervals.size() == 1)  return intervals;  // Sort the intervals based on the start values  Collections.sort(intervals, new IntervalComparator());  // Initialize the variables  Interval first = intervals.get(0);  int start = first.getStart();  int end = first.getEnd();  // Create a list for the merged intervals  ArrayList<Interval> result = new ArrayList<Interval>();  // Merge overlapping intervals  for (int i = 1; i < intervals.size(); i++) {  Interval current = intervals.get(i);  if (current.getStart() <= end) {  end = Math.max(current.getEnd(), end);  } else {  result.add(new Interval(start, end));  start = current.getStart();  end = current.getEnd();  }  }  result.add(new Interval(start, end));  return result;  }  }  class Interval  {  private int start;  private int end;  Interval() {  start = 0;  end = 0;  }  Interval(int s, int e)  {  start = s;  end = e;  }  public int getStart() {  return start;  }  public int getEnd() {  return end;  }  }  class IntervalComparator implements Comparator<Interval>  {  public int compare(Interval i1, Interval i2)  {  // Compare intervals based on start values  return i1.getStart() - i2.getStart();  }  } |

Output:

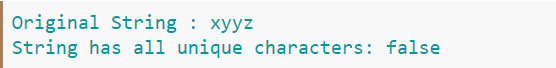
A white background with a black and white flag

Description automatically generated with medium confidence

**141.** Write a Java program to check if a given string has all distinct characters.  
Sample Output: Original String : xyyz  
String has all unique characters: false  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  public class Solution {  /\*\*  \* @param str: a string  \* @return: a boolean  \*/  public static boolean is\_Unique\_str(String str) {  // Convert the input string to a character array  char[] chars = str.toCharArray();    // Sort the character array in lexicographical order  Arrays.sort(chars);    // Check for repeated characters in the sorted array  for (int i = 1; i < chars.length; ++i) {  if (chars[i] == chars[i-1]) {  return false;  }  }    // If no repeated characters are found, the string is considered to have all unique characters  return true;  }  public static void main(String[] args) {  // Test case: Check if the string "xyyz" has all unique characters  // Note: You can change the value of the 'str' variable for different input strings.  String str = "xyyz";  System.out.println("Original String : " + str);    // Check if the string has all unique characters and print the result  System.out.println("String has all unique characters: " + is\_Unique\_str(str));  }  } |

Output:

**142.** Write a Java program to check if two strings are anagrams or not.  
According to Wikipedia "An anagram is a word or phrase formed by rearranging the letters of a different word or phrase, typically using all the original letters exactly once. For example, the word anagram can be rearranged into nag a ram, or the word binary into brainy."  
Sample Output: String-1 : wxyz  
String-2 : zyxw  
Check if two given strings are anagrams or not?: true  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*  public class Solution {  /\*\*  \* @param s: The first string  \* @param b: The second string  \* @return true or false  \*/  public static boolean anagram\_test(String str1, String str2) {  // Check if either input string is null  if (str1 == null || str2 == null) {  return false;  }  // Check if the lengths of the two strings are different  else if (str1.length() != str2.length()) {  return false;  }  // Check if both strings are empty (an edge case)  else if (str1.length() == 0 && str2.length() == 0) {  return true;  }    // Create an integer array to count character occurrences  int[] count = new int[256];    // Count character occurrences in both strings  for (int i = 0; i < str1.length(); i++) {  count[str1.charAt(i)]++;  count[str2.charAt(i)]--;  }  for (int num : count) {  if (num != 0) {  return false;  }  }    // If all checks pass, the strings are anagrams  return true;  }    public static void main(String[] args) {  // Test case: Check if the strings "wxyz" and "zyxw" are anagrams  String str1 = "wxyz";  String str2 = "zyxw";    // Print the original strings  System.out.println("String-1 : " + str1);  System.out.println("String-2 : " + str2);    // Check if the two given strings are anagrams and print the result  System.out.println("Check if two given strings are anagrams or not?: " + anagram\_test(str1, str2));  }  } |

Output:

A close-up of a white background

Description automatically generated**143.** Write a Java program to merge the two sorted linked lists.  
Sample Output:  
Merge Two Sorted ListsT:  
1 2 3 7 9 13 40  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*  public class Solution {  public static void main(String[] args) {  // Create two sorted linked lists  ListNode list1 = new ListNode(1);  list1.next = new ListNode(3);  list1.next.next = new ListNode(7);  list1.next.next.next = new ListNode(9);  list1.next.next.next.next = new ListNode(13);  ListNode list2 = new ListNode(2);  list2.next = new ListNode(40);    // Merge the two sorted lists and get the result  ListNode head = mergeTwoLists(list1, list2);  System.out.print("Merge Two Sorted Lists:\n");  while (head != null) {  System.out.print(head.val + " ");  head = head.next;  }  }  public static ListNode mergeTwoLists(ListNode list1, ListNode list2) {  // Create a new linked list for the merged result  ListNode head = new ListNode(0);  ListNode mlist = head;    // Merge the two lists while maintaining the order  while (list1 != null && list2 != null) {  if (list1.val < list2.val) {  mlist.next = new ListNode(list1.val);  mlist = mlist.next;  list1 = list1.next;  } else {  mlist.next = new ListNode(list2.val);  mlist = mlist.next;  list2 = list2.next;  }  }    // Append any remaining elements from list1  while (list1 != null) {  mlist.next = new ListNode(list1.val);  mlist = mlist.next;  list1 = list1.next;  }    // Append any remaining elements from list2  while (list2 != null) {  mlist.next = new ListNode(list2.val);  mlist = mlist.next;  list2 = list2.next;  }    // Skip the dummy head node and return the merged list  head = head.next;  return head;  }  }  class ListNode {  int val;  ListNode next;  ListNode(int val) {  this.val = val;  this.next = null;  }  } |

Output:

A close-up of a white background

Description automatically generated

**144.** Write a Java program to remove all occurrences of a specified value in a given array of integers. Return the updated array length.  
Sample Output:  
Original array: [1, 4, 6, 7, 6, 2]  
The length of the new array is: 4  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  public class Solution {  /\*\*  \* Remove all occurrences of a given element from an array and return the new length.  \* @param nums: A list of integers  \* @param element: An integer to be removed  \* @return: The new length after removing the element  \*/  public static int removeElement(int[] nums, int elem) {  int length = nums.length; // Get the initial length of the array  if (length == 0) return 0; // If the array is empty, return 0 (no changes)    int i = 0; // Initialize the index for the new array    // Iterate through the original array  for (int j = 0; j < length; j++) {  if (nums[j] != elem) {  // If the current element is not equal to the element to be removed  // Copy it to the new position in the array  nums[i] = nums[j];  i++;  }  }    // Replace elements beyond the new length with a sentinel value  if (i < length) nums[i] = '\0';    // Return the new length (the value of 'i')  return i;  }  public static void main(String[] args) {  int x = 6; // Element to be removed  int[] nums = {1, 4, 6, 7, 6, 2}; // Original array  System.out.println("Original array: " + Arrays.toString(nums));    // Call the removeElement function and print the new length  System.out.println("The length of the new array is: " + removeElement(nums, x));  }  } |

Output:

A close-up of a number

Description automatically generated**145.** Write a Java program to remove the nth element from the end of a given list.  
Sample Output:  
Original node:  
1 2 3 4 5  
After removing 2nd element from end:  
1 2 3 5  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  public class Solution {  public static void main(String[] args) {  // Create a linked list with five nodes (1, 2, 3, 4, 5)  ListNode h = new ListNode(1);  h.next = new ListNode(2);  h.next.next = new ListNode(3);  h.next.next.next = new ListNode(4);  h.next.next.next.next = new ListNode(5);    // Copy the original linked list to 'o' for display  ListNode o = h;  System.out.println("Original node:");    // Display the original linked list  while (o != null) {  System.out.print(o.val + " ");  o = o.next;  }    System.out.println("\nAfter removing 2nd element from end:");    // Call the removeNthFromEnd function and print the modified linked list  ListNode head = removeNthFromEnd(h, 2);    while (head != null) {  System.out.print(head.val + " ");  head = head.next;  }  }    public static ListNode removeNthFromEnd(ListNode head, int n) {  ListNode p = head;  int size = 0;    // Calculate the size of the linked list  while (p != null) {  size++;  p = p.next;  }    // Check if the element to remove is the first one  if (n == size) {  head = head.next;  } else {  int index = size - n;  ListNode t = head;    // Traverse to the node before the one to remove  while (index > 1) {  t = t.next;  index--;  }    // Update the 'next' reference to skip the node to remove  t.next = t.next.next;  }    return head; // Return the modified linked list  }  }  class ListNode {  int val;  ListNode next;  ListNode(int val) {  this.val = val;  this.next = null;  }  } |

Output:

A close-up of a white background

Description automatically generated

**146.** Write a Java program to convert an array of sorted items into a binary search tree. Maintain the minimal height of the tree.  
Sample Output:  
2  
1  
4  
6  
5  
3  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| public class Solution {  public static void main(String[] args) {  // Define an array of sorted integers  int[] arr = {1, 2, 3, 4, 5, 6};    // Convert the sorted array to a balanced binary search tree (BST)  TreeNode root = sortedArrayToBST(arr);    // Traverse the BST and print the values  traverseTree(root);  }  public static TreeNode sortedArrayToBST(int[] arr) {  if (arr.length == 0) return null;    // Call the recursive function for creating the BST  return creation(arr, 0, arr.length - 1);  }  private static TreeNode creation(int[] arr, int start, int end) {  TreeNode node = new TreeNode(0);    if (start == end - 1) {  // If the range contains two elements, create the nodes accordingly  node = new TreeNode(arr[start]);  node.right = new TreeNode(arr[end]);  } else if (start == end) {  // If the range contains a single element, create a node  return new TreeNode(arr[start]);  } else {  // Calculate the middle index of the range  int mid = (start + end) / 2;    // Set the value of the current node to the middle element  node.val = arr[mid];    // Recursively create left and right subtrees  node.left = creation(arr, start, mid - 1);  node.right = creation(arr, mid + 1, end);  }  return node;  }  private static void traverseTree(TreeNode root) {  // Post-order traversal of the BST (left, right, root)  if (root != null) {  traverseTree(root.left);  traverseTree(root.right);  System.out.println(root.val);  }  }  }  class TreeNode {  public int val;  public TreeNode left, right;  public TreeNode(int val) {  this.val = val;  this.left = this.right = null;  }  } |

Output:

A white background with a black and white flag

Description automatically generated with medium confidence

**147.** Write a Java program to find the number of bits required to flip to convert two given integers.  
Sample Output:  
2  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*  public class Solution {  public static void main(String[] args) {  // Test the bitSwapRequired function and print the result  System.out.println(bitSwapRequired(27, 23));  }  public static int bitSwapRequired(int x, int y) {  int ctr = 0; // Initialize a counter to keep track of bit differences  // XOR the two integers (x and y) to find differing bits  for (int z = x ^ y; z != 0; z = z >>> 1) {  // Right shift 'z' by 1 bit and check the least significant bit  ctr += z & 1; // If the least significant bit is 1, increment the counter  }  return ctr; // Return the total count of differing bits  }  } |

Output:

A white rectangular object with a black border

Description automatically generated

**148.** Write a Java program to find the index of the first unique character in a given string. Assume that there is at least one unique character in the string.  
Sample Output:  
Original String: wresource  
First unique character of the above: 0  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  public class Solution {  public static void main(String[] args) {  // Test the first\_Uniq\_Char function and print the result  String s = "wresource";  System.out.println("Original String: " + s);  System.out.println("First unique character of the above: " + first\_Uniq\_Char(s));  }  public static int first\_Uniq\_Char(String s) {  int[] freq = new int[256]; // Create an array to store character frequencies (assuming ASCII characters)  // Count the frequency of each character in the string  for (char c : s.toCharArray()) {  freq[c - 'a']++; // Increment the count at the corresponding index in the array  }  // Iterate through the string to find the index of the first unique character  for (int i = 0; i < s.length(); i++) {  if (freq[s.charAt(i) - 'a'] == 1) {  return i; // Return the index of the first character with a frequency of 1  }  }  return -1; // Return -1 if there are no unique characters  }  } |

Output:

A close up of a text

Description automatically generated

**149.** Write a Java program to check if a given string is a permutation of another given string.  
Sample Output:  
Original strings: xxyz yxzx  
true  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*;  public class Solution {  public static void main(String[] args) {  // Test the stringPermutation function and print the result  String str1 = "xxyz";  String str2 = "yxzx";  System.out.println("Original strings: " + str1 + " " + str2);  System.out.println(stringPermutation(str1, str2));  }  public static boolean stringPermutation(String str1, String str2) {  int[] arr = new int[500]; // Create an integer array to count character frequencies (assuming extended ASCII characters)  // Count the frequency of characters in the first string  for (int i = 0; i < str1.length(); i++) {  arr[(int) str1.charAt(i)] += 1;  }  // Decrement the count of characters in the second string  for (int i = 0; i < str2.length(); i++) {  arr[(int) str2.charAt(i)] -= 1;  }  // Check if all character counts are zero, indicating a permutation  for (int i = 0; i < arr.length; i++) {  if (arr[i] != 0) return false; // If any count is non-zero, return false  }  return true; // If all counts are zero, return true, indicating a permutation  }  } |

Output:



**150.** Write a Java program to test if a binary tree is a subtree of another binary tree.  
Sample Output:  
Original strings: xxyz yxzx  
true  
[Code:](https://www.w3resource.com/java-exercises/basic/java-basic-exercise-3.php)

|  |
| --- |
| import java.util.\*  public class Solution {  /\*\*  \* @param T1, T2: The roots of binary tree.  \* @return: True if T2 is a subtree of T1, or false.  \*/  public static boolean is\_Subtree(TreeNode T1, TreeNode T2) {  if (T2 == null) return true; // If T2 is null, it's always a subtree (base case)  if (T1 == null) return false; // If T1 is null but T2 is not, T2 can't be a subtree  if (is\_Same(T1, T2)) return true; // Check if the current subtrees are the same  if (is\_Subtree(T1.left, T2) || is\_Subtree(T1.right, T2)) return true; // Check left and right subtrees  return false; // If none of the above conditions match, T2 is not a subtree of T1  }  public static boolean is\_Same(TreeNode t1, TreeNode t2) {  if (t1 == null || t2 == null) {  return t1 == t2; // If one of the nodes is null, both should be null for them to be the same  } else if (t1.val != t2.val) {  return false; // If the values are different, the trees are not the same  } else {  // Recursively check the left and right subtrees for sameness  return is\_Same(t1.left, t2.left) && is\_Same(t1.right, t2.right);  }  }  public static void main(String[] args) {  // Create two binary trees  TreeNode t1 = new TreeNode(1);  TreeNode t2 = new TreeNode(2);  TreeNode t3 = new TreeNode(3);  TreeNode t4 = new TreeNode(4);  t1.left = t2;  t1.right = t3;  TreeNode n1 = new TreeNode(1);  TreeNode n2 = new TreeNode(2);  TreeNode n3 = new TreeNode(3);  TreeNode n4 = new TreeNode(4);  n1.left = n2;  n1.right = n3;  // Check if n1 is a subtree of t1 and print the result  System.out.println(is\_Subtree(t1, n1));  }  }  class TreeNode {  public int val;  public TreeNode left, right;  public TreeNode(int val) {  this.val = val;  this.left = this.right = null;  }  } |

Output:

A white rectangular object with a black border

Description automatically generated