Practice C++ Programming Statements after Foundation Training For Students of Complete Placement Preparatory Masterclass

Join Our Placement Preparation Masterclass on link below
https://talentbattle.in/prepare/placement-preparation

Example: How to Print an Integer entered by an user

```
import java.util.Scanner;

public class HelloWorld {

  public static void main(String[] args) {

    // Creates a reader instance which takes
    // input from standard input - keyboard
    Scanner reader = new Scanner(System.in);
    System.out.print("Enter a number: ");

    // nextInt() reads the next integer from the keyboard
    int number = reader.nextInt();

    // println() prints the following line to the output screen
    System.out.println("You entered: " + number);
  }
}
```

****

Example: Program to Add Two Integers

```
class Main {

 public static void main(String[] args) {

  System.out.println("Enter two numbers");
  int first = 10;
  int second = 20;

  System.out.println(first + " " + second);
```

```
    // add two numbers
    int sum = first + second;

    System.out.println("The sum is: " + sum);
  }
}
```

****

Example: Multiply Two Floating-Point Numbers

```
public class MultiplyTwoNumbers {

  public static void main(String[] args) {

    float first = 1.5f;

    float second = 2.0f;

    float product = first * second;

    System.out.println("The product is: " + product);

  }
}
```

****

Example: Find ASCII value of a character

```
public class AsciiValue {

  public static void main(String[] args) {

    char ch = 'a';

    int ascii = ch;
    // You can also cast char to int
    int castAscii = (int) ch;
```

```
    System.out.println("The ASCII value of " + ch + " is: " + ascii);
    System.out.println("The ASCII value of " + ch + " is: " + castAscii);
  }
}
```

****

Example: Compute Quotient and Remainder

```
public class QuotientRemainder {

  public static void main(String[] args) {

    int dividend = 25, divisor = 4;

    int quotient = dividend / divisor;
    int remainder = dividend % divisor;

    System.out.println("Quotient = " + quotient);
    System.out.println("Remainder = " + remainder);
  }
}
```

****

Example 1: Swap two numbers using temporary variable

```
public class SwapNumbers {

    public static void main(String[] args) {

        float first = 1.20f, second = 2.45f;

        System.out.println("--Before swap--");
        System.out.println("First number = " + first);
```

```
        System.out.println("Second number = " + second);

        // Value of first is assigned to temporary
        float temporary = first;

        // Value of second is assigned to first
        first = second;

        // Value of temporary (which contains the initial value of first) is assigned to second
        second = temporary;

        System.out.println("--After swap--");
        System.out.println("First number = " + first);
        System.out.println("Second number = " + second);
    }
}
```

****

Example 2: Swap two numbers without using temporary variable

```
public class SwapNumbers {

    public static void main(String[] args) {

        float first = 12.0f, second = 24.5f;

        System.out.println("--Before swap--");
        System.out.println("First number = " + first);
        System.out.println("Second number = " + second);

        first = first - second;
        second = first + second;
        first = second - first;
```

```
    System.out.println("--After swap--");
    System.out.println("First number = " + first);
    System.out.println("Second number = " + second);
  }
}
```

****

Example 1: Check whether a number is even or odd using if...else statement

```
import java.util.Scanner;

public class EvenOdd {

  public static void main(String[] args) {

    Scanner reader = new Scanner(System.in);

    System.out.print("Enter a number: ");
    int num = reader.nextInt();

    if(num % 2 == 0)
      System.out.println(num + " is even");
    else
      System.out.println(num + " is odd");
  }
}
```

****

Example 2: Check whether a number is even or odd using ternary operator

```
import java.util.Scanner;

public class EvenOdd {

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

```
        Scanner reader = new Scanner(System.in);

        System.out.print("Enter a number: ");
        int num = reader.nextInt();

        String evenOdd = (num % 2 == 0) ? "even" : "odd";

        System.out.println(num + " is " + evenOdd);

    }
}
```

****

Example 1: Check whether an alphabet is vowel or consonant using if..else statement

```
public class VowelConsonant {

  public static void main(String[] args) {

      char ch = 'i';

      if(ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u' )
        System.out.println(ch + " is vowel");
      else
        System.out.println(ch + " is consonant");

    }
}
```

****

Example 2: Check whether an alphabet is vowel or consonant using switch statement

```
public class VowelConsonant {

  public static void main(String[] args) {


    char ch = 'z';


    switch (ch) {
      case 'a':
      case 'e':
      case 'i':
      case 'o':
      case 'u':
        System.out.println(ch + " is vowel");
        break;
      default:
        System.out.println(ch + " is consonant");
    }
  }
}
```

****

Example 1: Find Largest Among three numbers using if..else statement

```
public class Largest {

  public static void main(String[] args) {

    double n1 = -4.5, n2 = 3.9, n3 = 2.5;


    if( n1 >= n2 && n1 >= n3)
      System.out.println(n1 + " is the largest number.");


    else if(n2 >= n1 && n2 >= n3)
      System.out.println(n2 + " is the largest number.");
```

```
    else
      System.out.println(n3 + " is the largest number.");
  }
}
```

****

Example 2: Find the largest number among three using nested if..else statement

```
public class Largest {

  public static void main(String[] args) {

    double n1 = -4.5, n2 = 3.9, n3 = 5.5;

    if(n1 >= n2) {
      if(n1 >= n3)
        System.out.println(n1 + " is the largest number.");
      else
        System.out.println(n3 + " is the largest number.");
    } else {
      if(n2 >= n3)
        System.out.println(n2 + " is the largest number.");
      else
        System.out.println(n3 + " is the largest number.");
    }
  }
}
```

****

Example: Java Program to Find Roots of a Quadratic Equation

```
public class Main {
```

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

 // value a, b, and c
 double a = 2.3, b = 4, c = 5.6;
 double root1, root2;

 // calculate the determinant (b2 - 4ac)
 double determinant = b * b - 4 * a * c;

 // check if determinant is greater than 0
 if (determinant > 0) {

  // two real and distinct roots
  root1 = (-b + Math.sqrt(determinant)) / (2 * a);
  root2 = (-b - Math.sqrt(determinant)) / (2 * a);

  System.out.format("root1 = %.2f and root2 = %.2f", root1, root2);
 }

 // check if determinant is equal to 0
 else if (determinant == 0) {

  // two real and equal roots
  // determinant is equal to 0
  // so -b + 0 == -b
  root1 = root2 = -b / (2 * a);
  System.out.format("root1 = root2 = %.2f;", root1);
 }

 // if determinant is less than zero
 else {

  // roots are complex number and distinct
```

```
    double real = -b / (2 * a);
    double imaginary = Math.sqrt(-determinant) / (2 * a);
    System.out.format("root1 = %.2f+%.2fi", real, imaginary);
    System.out.format("\nroot2 = %.2f-%.2fi", real, imaginary);
  }
 }
}
```

****

Example: Java Program to Check a Leap Year

```
public class Main {

 public static void main(String[] args) {

  // year to be checked
  int year = 1996;
  boolean leap = false;

  // if the year is divided by 4
  if (year % 4 == 0) {

   // if the year is century
   if (year % 100 == 0) {

    // if year is divided by 400
    // then it is a leap year
    if (year % 400 == 0)
    leap = true;
    else
     leap = false;
   }
```

```
    // if the year is not century
    else
      leap = true;
  }

  else
    leap = false;

  if (leap)
    System.out.println(year + " is a leap year.");
  else
    System.out.println(year + " is not a leap year.");
 }
}
```

****

Example: Check if a Number is Positive or Negative using if else

```
public class PositiveNegative {

  public static void main(String[] args) {

    double number = 12.3;

    // true if number is less than 0
    if (number < 0.0)
      System.out.println(number + " is a negative number.");

    // true if number is greater than 0
    else if( number > 0.0)
      System.out.println(number + " is a positive number.");

    // if both test expression is evaluated to false
    else
```

```
        System.out.println(number + " is 0.");
    }
}
```

****

Example 1: Java Program to Check Alphabet using if else

```
public class Alphabet {

    public static void main(String[] args) {

        char c = '*';

        if( (c >= 'a' && c <= 'z') || (c >= 'A' && c <= 'Z') )
            System.out.println(c + " is an alphabet.");
        else
            System.out.println(c + " is not an alphabet.");
    }
}
```

****

Example 2: Java Program to Check Alphabet using ternary operator

```
public class Alphabet {

    public static void main(String[] args) {

        char c = 'A';

        String output = (c >= 'a' && c <= 'z') || (c >= 'A' && c <= 'Z')
                ? c + " is an alphabet."
                : c + " is not an alphabet.";

        System.out.println(output);
```

```
  }
}
```

****

Example 3: Java Program to Check Alphabet using isAlphabetic() Method

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

  // declare a variable
  char c = 'a';

  // checks if c is an alphabet
  if (Character.isAlphabetic(c)) {
   System.out.println(c + " is an alphabet.");
  }
  else {
   System.out.println(c + " is not an alphabet.");
  }
 }
}
```

****

Example 1: Sum of Natural Numbers using for loop

```
public class SumNatural {

  public static void main(String[] args) {

    int num = 100,  sum = 0;


    for(int i = 1; i <= num; ++i)
    {
      // sum = sum + i;
      sum += i;
    }
```

```
    System.out.println("Sum = " + sum);
  }
}
```

****

Example 2: Sum of Natural Numbers using while loop

```
public class SumNatural {

  public static void main(String[] args) {

    int num = 50, i = 1, sum = 0;

    while(i <= num)
    {
      sum += i;
      i++;
    }

    System.out.println("Sum = " + sum);
  }
}
```

****

Example 1: Find Factorial of a number using for loop

```
public class Factorial {

  public static void main(String[] args) {

    int num = 10;
    long factorial = 1;
    for(int i = 1; i <= num; ++i)
    {
```

```
        // factorial = factorial * i;
        factorial *= i;
      }
      System.out.printf("Factorial of %d = %d", num, factorial);
   }
}
```

****

Example 2: Find Factorial of a number using BigInteger

```
import java.math.BigInteger;

public class Factorial {

   public static void main(String[] args) {

      int num = 30;
      BigInteger factorial = BigInteger.ONE;
      for(int i = 1; i <= num; ++i)
      {
         // factorial = factorial * i;
         factorial = factorial.multiply(BigInteger.valueOf(i));
      }
      System.out.printf("Factorial of %d = %d", num, factorial);
   }
}
```

****

Example 3: Find Factorial of a number using while loop

```
public class Factorial {

   public static void main(String[] args) {

      int num = 5, i = 1;
      long factorial = 1;
```

```
    while(i <= num)
    {
      factorial *= i;
      i++;
    }
    System.out.printf("Factorial of %d = %d", num, factorial);
  }
}
```

****

Example 1: Generate Multiplication Table using for loop

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

    int num = 5;
    for(int i = 1; i <= 10; ++i)
    {
      System.out.printf("%d * %d = %d \n", num, i, num * i);
    }
  }
}
```

****

Example 2: Generate Multiplication Table using while loop

```
public class MultiplicationTable {

  public static void main(String[] args) {

    int num = 9, i = 1;
    while(i <= 10)
    {
      System.out.printf("%d * %d = %d \n", num, i, num * i);
      i++;
```

```
    }
  }
}
```

****

Example: Display Fibonacci Series Using for Loop

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

    int n = 10,  firstTerm = 0, secondTerm = 1;
    System.out.println("Fibonacci Series till " + n + " terms:");

    for (int i = 1; i <= n; ++i) {
      System.out.print(firstTerm + ", ");

      // compute the next term
      int nextTerm = firstTerm + secondTerm;
      firstTerm = secondTerm;
      secondTerm = nextTerm;
    }
  }
}
```

****

Example 2: Display Fibonacci series using while loop

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

    int i = 1, n = 10, firstTerm = 0, secondTerm = 1;
    System.out.println("Fibonacci Series till " + n + " terms:");

    while (i <= n) {
      System.out.print(firstTerm + ", ");
```

```
    int nextTerm = firstTerm + secondTerm;
    firstTerm = secondTerm;
    secondTerm = nextTerm;

    i++;
   }
  }
}
```

****

Example 3: Display Fibonacci series up to a given number

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

   int n = 100,  firstTerm = 0, secondTerm = 1;

   System.out.println("Fibonacci Series Upto " + n + ": ");

   while (firstTerm <= n) {
    System.out.print(firstTerm + ", ");

    int nextTerm = firstTerm + secondTerm;
    firstTerm = secondTerm;
    secondTerm = nextTerm;

   }
  }
}
```

****

Example 1: Find GCD of two numbers using for loop and if statement

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

    // find GCD between n1 and n2
    int n1 = 81, n2 = 153;

    // initially set to gcd
    int gcd = 1;

    for (int i = 1; i <= n1 && i <= n2; ++i) {

      // check if i perfectly divides both n1 and n2
      if (n1 % i == 0 && n2 % i == 0)
        gcd = i;
    }

    System.out.println("GCD of " + n1 +" and " + n2 + " is " + gcd);
  }
}
```

****

Example 2: Find GCD of two numbers using while loop and if else statement

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

    // find GCD between n1 and n2
    int n1 = 81, n2 = 153;

    while(n1 != n2) {

      if(n1 > n2) {
        n1 -= n2;
```

```
    }

    else {
      n2 -= n1;
    }
   }

  System.out.println("GCD: " + n1);
 }
}
```

****

Example 3: GCD for both positive and negative numbers

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

  int n1 = 81, n2 = -153;

  // Always set to positive
  n1 = ( n1 > 0) ? n1 : -n1;
  n2 = ( n2 > 0) ? n2 : -n2;
  while(n1 != n2) {

   if(n1 > n2) {
     n1 -= n2;
   }

   else {
     n2 -= n1;
   }
  }
```

```
    System.out.println("GCD: " + n1);
  }
}
```

****

Example 1: LCM using while Loop and if Statement

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

    int n1 = 72, n2 = 120, lcm;

    // maximum number between n1 and n2 is stored in lcm
    lcm = (n1 > n2) ? n1 : n2;

    // Always true
    while(true) {
      if( lcm % n1 == 0 && lcm % n2 == 0 ) {
        System.out.printf("The LCM of %d and %d is %d.", n1, n2, lcm);
        break;
      }
      ++lcm;
    }
  }
}
```

****

Example 2: Calculate LCM using GCD

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

    int n1 = 72, n2 = 120, gcd = 1;

    for(int i = 1; i <= n1 && i <= n2; ++i) {
```

```
    // Checks if i is factor of both integers
    if(n1 % i == 0 && n2 % i == 0)
      gcd = i;
   }

   int lcm = (n1 * n2) / gcd;
   System.out.printf("The LCM of %d and %d is %d.", n1, n2, lcm);
  }
}
```

****

Example 1: Display uppercased alphabet using for loop

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

   char c;

   for(c = 'A'; c <= 'Z'; ++c)
    System.out.print(c + " ");
   }
}
```

****

Example 2: Display lowercase alphabet using for loop

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

   char c;

   for(c = 'a'; c <= 'z'; ++c)
    System.out.print(c + " ");
   }
}
```

****

Example 1: Count Number of Digits in an Integer using while loop

```
public class Main {

 public static void main(String[] args) {


  int count = 0, num = 0003452;


  while (num != 0) {
   // num = num/10
   num /= 10;

   ++count;
  }


  System.out.println("Number of digits: " + count);
 }
}
```

****

Example 2: Count Number of Digits in an Integer using for loop

```
public class Main {

 public static void main(String[] args) {


  int count = 0, num = 123456;


  for (; num != 0; num /= 10, ++count) {
  }


  System.out.println("Number of digits: " + count);
 }
}
```

****

Example 1: Reverse a Number using a while loop in Java

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

```
    int num = 1234,  reversed = 0;

    // run loop until num becomes 0
    while(num != 0) {

      // get last digit from num
      int digit = num % 10;
      reversed = reversed * 10 + digit;

      // remove the last digit from num
      num /= 10;
    }

    System.out.println("Reversed Number: " + reversed);
  }
}
```

****

Example 2: Reverse a number using a for loop in Java

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

    int num = 1234567,  reversed = 0;

    for(;num != 0; num /= 10) {
      int digit = num % 10;
      reversed = reversed * 10 + digit;
    }

    System.out.println("Reversed Number: " + reversed);
  }
}
```

****

Example 1: Calculate power of a number using a while loop

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

    int base = 3, exponent = 4;

    long result = 1;

    while (exponent != 0) {
      result *= base;
      --exponent;
    }

    System.out.println("Answer = " + result);
  }
}
```

****

Example 2: Calculate the power of a number using a for loop

```
class Main {
  public static void main(String[] args) {
    int base = 3, exponent = 4;

    long result = 1;

    for (; exponent != 0; --exponent) {
      result *= base;
    }

    System.out.println("Answer = " + result);
  }
}
```

****

Example 3: Calculate the power of a number using pow() function

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

    int base = 3, exponent = -4;
    double result = Math.pow(base, exponent);

    System.out.println("Answer = " + result);
  }
}
```

****

Example 4: Compute Power of Negative Number

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

    // negative number
    int base = -3, exponent = 2;

    double result = Math.pow(base, exponent);
    System.out.println("Answer = " + result);
  }
}
```

****

Example 1: Java Program to Check Palindrome String

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

    String str = "Radar", reverseStr = "";
```

```
    int strLength = str.length();

    for (int i = (strLength - 1); i >= 0; --i) {
      reverseStr = reverseStr + str.charAt(i);
    }

    if (str.toLowerCase().equals(reverseStr.toLowerCase())) {
      System.out.println(str + " is a Palindrome String.");
    }
    else {
      System.out.println(str + " is not a Palindrome String.");
    }
  }
}
```

****

Example 2: Java Program to Check Palindrome Number

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

    int num = 3553,  reversedNum = 0, remainder;

    // store the number to originalNum
    int originalNum = num;

    // get the reverse of originalNum
    // store it in variable
    while (num != 0) {
    remainder = num % 10;
      reversedNum = reversedNum * 10 + remainder;
      num /= 10;
    }
```

```
    // check if reversedNum and originalNum are equal
    if (originalNum == reversedNum) {
      System.out.println(originalNum + " is Palindrome.");
    }
    else {
      System.out.println(originalNum + " is not Palindrome.");
    }
  }
}
```

****

Example 1: Program to Check Prime Number using a for loop

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

    int num = 29;
    boolean flag = false;
    for (int i = 2; i <= num / 2; ++i) {
      // condition for nonprime number
      if (num % i == 0) {
        flag = true;
        break;
      }
    }

    if (!flag)
      System.out.println(num + " is a prime number.");
    else
      System.out.println(num + " is not a prime number.");
  }
}
```

****

Example 2: Program to Check Prime Number using a while loop

```
public class Main {

 public static void main(String[] args) {

  int num = 33, i = 2;
  boolean flag = false;
  while (i <= num / 2) {
   // condition for nonprime number
   if (num % i == 0) {
    flag = true;
    break;
   }

   ++i;
  }

  if (!flag)
   System.out.println(num + " is a prime number.");
  else
   System.out.println(num + " is not a prime number.");
 }
}
```

****

Example: Prime Numbers Between Two Integers

```
public class Prime {

  public static void main(String[] args) {

    int low = 20, high = 50;
```

```
    while (low < high) {
      if(checkPrimeNumber(low))
        System.out.print(low + " ");
      ++low;
    }
  }

  public static boolean checkPrimeNumber( int num) {
    boolean flag = true;

    for(int i = 2; i <= num/2; ++i) {

      if(num % i == 0) {
        flag = false;
        break;
      }
    }

    return flag;
  }
}
```

****

Example: Armstrong Numbers Between Two Integers

```
public class Armstrong {

  public static void main(String[] args) {

    int low = 999, high = 99999;

    for(int number = low + 1; number < high; ++number) {
```

```
        if (checkArmstrong(number))
            System.out.print(number + " ");
        }
    }

    public static boolean checkArmstrong(int num) {
        int digits = 0;
        int result = 0;
        int originalNumber = num;

        // number of digits calculation
        while (originalNumber != 0) {
            originalNumber /= 10;
            ++digits;
        }

        originalNumber = num;

        // result contains sum of nth power of its digits
        while (originalNumber != 0) {
            int remainder = originalNumber % 10;
            result += Math.pow(remainder, digits);
            originalNumber /= 10;
        }

        if (result == num)
            return true;

        return false;
    }
}
```

****

Example: Represent a number as Sum of Two Prime Numbers

```
public class Main {

 public static void main(String[] args) {
  int number = 34;
  boolean flag = false;
  for (int i = 2; i <= number / 2; ++i) {

   // condition for i to be a prime number
   if (checkPrime(i)) {

    // condition for n-i to be a prime number
    if (checkPrime(number - i)) {

     // n = primeNumber1 + primeNumber2
     System.out.printf("%d = %d + %d\n", number, i, number - i);
     flag = true;
    }

   }
  }

  if ( !flag)
   System.out.println(number + " cannot be expressed as the sum of two prime numbers.");
 }

 // Function to check prime number
 static boolean checkPrime(int num) {
  boolean isPrime = true;

  for (int i = 2; i <= num / 2; ++i) {
   if (num % i == 0) {
    isPrime = false;
```

```
      break;
    }
   }

   return isPrime;
  }
}
```

****

Example: Sum of Natural Numbers Using Recursion

```
public class AddNumbers {

   public static void main(String[] args) {
      int number = 20;
      int sum = addNumbers(number);
      System.out.println("Sum = " + sum);
   }

   public static int addNumbers(int num) {
      if (num != 0)
         return num + addNumbers(num - 1);
      else
         return num;
   }
}
```

****

Example: Factorial of a Number Using Recursion

```
public class Factorial {

   public static void main(String[] args) {
      int num = 6;
      long factorial = multiplyNumbers(num);
      System.out.println("Factorial of " + num + " = " + factorial);
```

```
  }
  public static long multiplyNumbers(int num)
  {
    if (num >= 1)
      return num * multiplyNumbers(num - 1);
    else
      return 1;
  }
}
```

****

Example: GCD of Two Numbers using Recursion

```
public class GCD {

  public static void main(String[] args) {
    int n1 = 366,  n2 = 60;
    int hcf = hcf(n1, n2);

    System.out.printf("G.C.D of %d and %d is %d.", n1, n2, hcf);
  }

  public static int hcf(int n1, int n2)
  {
    if (n2 != 0)
      return hcf(n2, n1 % n2);
    else
      return n1;
  }
}
```

****

Example 1: Binary to Decimal Conversion Using Custom Method

```
class Main {
```

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

    // binary number
    long num = 110110111;

    // call method by passing the binary number
    int decimal = convertBinaryToDecimal(num);

    System.out.println("Binary to Decimal");
    System.out.println(num + " = " + decimal);
  }

  public static int convertBinaryToDecimal(long num) {
    int decimalNumber = 0, i = 0;
    long remainder;

    while (num != 0) {
      remainder = num % 10;
      num /= 10;
      decimalNumber += remainder * Math.pow(2, i);
      ++i;
    }

    return decimalNumber;
  }
}
```

****

Example 2: Binary to Decimal Conversion Using parseInt()

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

    // binary number
```

```
  String binary = "01011011";

  // convert to decimal
  int decimal = Integer.parseInt(binary, 2);
  System.out.println(binary + " in binary = " + decimal + " in decimal.");
 }
}
```

****

Example 3: Decimal to Binary Conversion using Custom Method

```
class Main {

 public static void main(String[] args) {

  // decimal number
  int num = 19;
  System.out.println("Decimal to Binary");

  // call method to convert to binary
  long binary = convertDecimalToBinary(num);

  System.out.println("\n" + num + " = " + binary);
  }

 public static long convertDecimalToBinary(int n) {

  long binaryNumber = 0;

  int remainder, i = 1, step = 1;

  while (n!=0) {
   remainder = n % 2;

    System.out.println("Step " + step++ + ": " + n + "/2");

    System.out.println("Quotient = " + n/2 + ", Remainder = " + remainder);
```

```
    n /= 2;

    binaryNumber += remainder * i;
    i *= 10;
  }

  return binaryNumber;
  }
}
```

****

Example 4: Decimal to Binary Conversion using toBinaryString()

We can also use the toBinaryString() method of the Integer class to convert a decimal number into binary.

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

    // decimal number
    int decimal = 91;

    // convert decimal to binary
    String binary = Integer.toBinaryString(decimal);
    System.out.println(decimal + " in decimal = " + binary + " in binary.");
  }
}
```

****

Example 1: Program to Convert Decimal to Octal

```
public class DecimalOctal {

  public static void main(String[] args) {
    int decimal = 78;
    int octal = convertDecimalToOctal(decimal);
    System.out.printf("%d in decimal = %d in octal", decimal, octal);
  }
```

```
  public static int convertDecimalToOctal(int decimal)
  {
     int octalNumber = 0, i = 1;

     while (decimal != 0)
     {
        octalNumber += (decimal % 8) * i;
        decimal /= 8;

        i *= 10;
     }

     return octalNumber;
  }
}
```

****

Example 2: Program to Convert Octal to Decimal

```
public class OctalDecimal {

  public static void main(String[] args) {
     int octal = 116;
     int decimal = convertOctalToDecimal(octal);
     System.out.printf("%d in octal = %d in decimal", octal, decimal);
  }


  public static int convertOctalToDecimal(int octal)
  {
     int decimalNumber = 0, i = 0;

     while(octal != 0)
     {
        decimalNumber += (octal % 10) * Math.pow(8, i);
        ++i;
```

```
    octal/=10;
    }

    return decimalNumber;
  }
}
```

****

Example 1: Program to Convert Binary to Octal

In this program, we will first convert binary number to decimal. Then, the decimal number is converted to octal.

```
class Main {
  public static void main(String[] args) {
    long binary = 101001;
    int octal = convertBinarytoOctal(binary);
    System.out.println(binary + " in binary = " + octal + " in octal");
  }

  public static int convertBinarytoOctal(long binaryNumber) {
    int octalNumber = 0, decimalNumber = 0, i = 0;

    while (binaryNumber != 0) {
      decimalNumber += (binaryNumber % 10) * Math.pow(2, i);
      ++i;
      binaryNumber /= 10;
    }

    i = 1;

    while (decimalNumber != 0) {
      octalNumber += (decimalNumber % 8) * i;
      decimalNumber /= 8;
      i *= 10;
```

```
  }

  return octalNumber;
 }
}
```

****

Example 2: Program to Convert Octal to Binary

In this program, the octal number to decimal to decimal at first. Then, the decimal number is converted to binary number.

```
class Main {
 public static void main(String[] args) {
  int octal = 67;
  long binary = convertOctalToBinary(octal);
  System.out.println(octal + " in octal = " + binary + " in binary");
 }

 public static long convertOctalToBinary(int octalNumber) {
  int decimalNumber = 0, i = 0;
  long binaryNumber = 0;

  while (octalNumber != 0) {
   decimalNumber += (octalNumber % 10) * Math.pow(8, i);
   ++i;
   octalNumber /= 10;
  }

  i = 1;

  while (decimalNumber != 0) {
   binaryNumber += (decimalNumber % 2) * i;
   decimalNumber /= 2;
   i *= 10;
```

```
  }

  return binaryNumber;
 }
}
```

****

Example: Reverse a Sentence Using Recursion

```
public class Reverse {

 public static void main(String[] args) {
  String sentence = "Go work";
  String reversed = reverse(sentence);
  System.out.println("The reversed sentence is: " + reversed);
 }

 public static String reverse(String sentence) {
  if (sentence.isEmpty())
   return sentence;

  return reverse(sentence.substring(1)) + sentence.charAt(0);
 }
}
```

****

Example: Simple Calculator using Java switch Statement

```
import java.util.Scanner;

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

  char operator;
  Double number1, number2, result;
```

```
// create an object of Scanner class
Scanner input = new Scanner(System.in);

// ask users to enter operator
System.out.println("Choose an operator: +, -, *, or /");
operator = input.next().charAt(0);

// ask users to enter numbers
System.out.println("Enter first number");
number1 = input.nextDouble();

System.out.println("Enter second number");
number2 = input.nextDouble();

switch (operator) {

 // performs addition between numbers
 case '+':
  result = number1 + number2;
  System.out.println(number1 + " + " + number2 + " = " + result);
  break;

 // performs subtraction between numbers
 case '-':
  result = number1 - number2;
  System.out.println(number1 + " - " + number2 + " = " + result);
  break;

 // performs multiplication between numbers
 case '*':
  result = number1 * number2;
  System.out.println(number1 + " * " + number2 + " = " + result);
  break;
```

```
    // performs division between numbers
    case '/':
      result = number1 / number2;
      System.out.println(number1 + " / " + number2 + " = " + result);
      break;

    default:
      System.out.println("Invalid operator!");
      break;
  }

  input.close();
 }
}
```

****

Example: Program to calculate power using recursion

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

  int base = 3, powerRaised = 4;
  int result = power(base, powerRaised);

  System.out.println(base + "^" + powerRaised + "=" + result);
 }

 public static int power(int base, int powerRaised) {
  if (powerRaised != 0) {

   // recursive call to power()
   return (base * power(base, powerRaised - 1));
  }
  else {
```

```
    return 1;
  }
 }
}
```

****

Example: Program to Calculate Average Using Arrays

```
public class Average {

  public static void main(String[] args) {
    double[] numArray = {45.3, 67.5, -45.6, 20.34, 33.0, 45.6};
    double sum = 0.0;

    for (double num: numArray) {
      sum += num;
    }

    double average = sum / numArray.length;
    System.out.format("The average is: %.2f", average);
  }
}
```

****

Example: Find the largest element in an array

```
public class Largest {

  public static void main(String[] args) {
    double[] numArray = {23.4, -34.5, 50.0, 33.5, 55.5, 43.7, 5.7, -66.5};
    double largest = numArray[0];

    for (double num: numArray) {
      if(largest < num)
        largest = num;
    }
```

```
    System.out.format("Largest element = %.2f", largest);
  }
}
```

****

Example: Program to Calculate Standard Deviation

```
public class StandardDeviation {

  public static void main(String[] args) {
    double[] numArray = { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 };
    double SD = calculateSD(numArray);

    System.out.format("Standard Deviation = %.6f", SD);
  }

  public static double calculateSD(double numArray[])
  {
    double sum = 0.0, standardDeviation = 0.0;
    int length = numArray.length;

    for(double num : numArray) {
      sum += num;
    }

    double mean = sum/length;

    for(double num: numArray) {

      standardDeviation += Math.pow(num - mean, 2);
    }

    return Math.sqrt(standardDeviation/length);
  }
}
```

****

Example: Program to Add Two Matrices

```
public class AddMatrices {

  public static void main(String[] args) {
    int rows = 2, columns = 3;
    int[][] firstMatrix = { {2, 3, 4}, {5, 2, 3} };
    int[][] secondMatrix = { {-4, 5, 3}, {5, 6, 3} };

    // Adding Two matrices
    int[][] sum = new int[rows][columns];
    for(int i = 0; i < rows; i++) {
      for (int j = 0; j < columns; j++) {
        sum[i][j] = firstMatrix[i][j] + secondMatrix[i][j];
      }
    }

    // Displaying the result
    System.out.println("Sum of two matrices is: ");
    for(int[] row : sum) {
      for (int column : row) {
        System.out.print(column + "    ");
      }
      System.out.println();
    }
  }
}
```

****

Example: Program to Multiply Two Matrices

```
public class MultiplyMatrices {

  public static void main(String[] args) {
    int r1 = 2, c1 = 3;
```

```
    int r2 = 3, c2 = 2;
    int[][] firstMatrix = { {3, -2, 5}, {3, 0, 4} };
    int[][] secondMatrix = { {2, 3}, {-9, 0}, {0, 4} };

    // Mutliplying Two matrices
    int[][] product = new int[r1][c2];
    for(int i = 0; i < r1; i++) {
      for (int j = 0; j < c2; j++) {
        for (int k = 0; k < c1; k++) {
          product[i][j] += firstMatrix[i][k] * secondMatrix[k][j];
        }
      }
    }

    // Displaying the result
    System.out.println("Sum of two matrices is: ");
    for(int[] row : product) {
      for (int column : row) {
        System.out.print(column + "    ");
      }
      System.out.println();
    }
  }
}
```

****

Example: Program to Multiply Two Matrices using a Function

```
public class MultiplyMatrices {

  public static void main(String[] args) {
    int r1 = 2, c1 = 3;
    int r2 = 3, c2 = 2;
    int[][] firstMatrix = { {3, -2, 5}, {3, 0, 4} };
    int[][] secondMatrix = { {2, 3}, {-9, 0}, {0, 4} };
```

```
        // Mutliplying Two matrices
        int[][] product = multiplyMatrices(firstMatrix, secondMatrix, r1, c1, c2);

        // Displaying the result
        displayProduct(product);
    }

    public static int[][] multiplyMatrices(int[][] firstMatrix, int[][] secondMatrix, int r1, int c1, int c2) {
        int[][] product = new int[r1][c2];
        for(int i = 0; i < r1; i++) {
            for (int j = 0; j < c2; j++) {
                for (int k = 0; k < c1; k++) {
                    product[i][j] += firstMatrix[i][k] * secondMatrix[k][j];
                }
            }
        }

        return product;
    }

    public static void displayProduct(int[][] product) {
        System.out.println("Product of two matrices is: ");
        for(int[] row : product) {
            for (int column : row) {
                System.out.print(column + "    ");
            }
            System.out.println();
        }
    }
}
```

****

Example: Program to Find Transpose of a Matrix

```
public class Transpose {

  public static void main(String[] args) {
    int row = 2, column = 3;
    int[][] matrix = { {2, 3, 4}, {5, 6, 4} };

    // Display current matrix
    display(matrix);

    // Transpose the matrix
    int[][] transpose = new int[column][row];
    for(int i = 0; i < row; i++) {
      for (int j = 0; j < column; j++) {
        transpose[j][i] = matrix[i][j];
      }
    }

    // Display transposed matrix
    display(transpose);
  }

  public static void display(int[][] matrix) {
    System.out.println("The matrix is: ");
    for(int[] row : matrix) {
      for (int column : row) {
        System.out.print(column + "    ");
      }
      System.out.println();
    }
  }
}
```

****

Example: Find Frequency of Character

```
public class Frequency {

  public static void main(String[] args) {
    String str = "This website is awesome.";
    char ch = 'e';
    int frequency = 0;

    for(int i = 0; i < str.length(); i++) {
      if(ch == str.charAt(i)) {
        ++frequency;
      }
    }

    System.out.println("Frequency of " + ch + " = " + frequency);
  }
}
```

****

Example: Program to count vowels, consonants, digits, and spaces

```
class Main {

 public static void main(String[] args) {
  String line = "This website is aw3som3.";
  int vowels = 0, consonants = 0, digits = 0, spaces = 0;

  line = line.toLowerCase();
  for (int i = 0; i < line.length(); ++i) {
   char ch = line.charAt(i);

   // check if character is any of a, e, i, o, u
   if (ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u') {
    ++vowels;
   }
```

```
    // check if character is in between a to z
    else if((ch >= 'a' && ch <= 'z')) {
      ++consonants;
    }

    // check if character is in between 0 to 9
    else if(ch >= '0' && ch <= '9') {
      ++digits;
    }

    // check if character is a white space
    else if(ch == ' ') {
      ++spaces;
    }
  }

  System.out.println("Vowels: " + vowels);
  System.out.println("Consonants: " + consonants);
  System.out.println("Digits: " + digits);
  System.out.println("White spaces: " + spaces);
 }
}
```

****

Example: Program to Sort Strings in Dictionary Order

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

    String[] words = { "Ruby", "C", "Python", "Java" };

    for(int i = 0; i < 3; ++i) {
      for (int j = i + 1; j < 4; ++j) {

        if (words[i].compareTo(words[j]) > 0) {

          // swap words[i] with words[j[
          String temp = words[i];
          words[ i] = words[j];
          words[j] = temp;
        }
      }
    }

    System.out.println("In lexicographical order:");

    for(int i = 0; i < 4; i++) {
      System.out.println(words[i]);
    }
  }
}
```

****

Example: Add Two Complex Numbers

```
public class Complex {

  double real;
  double imag;

  public Complex(double real, double imag) {
     this.real = real;
     this.imag = imag;
  }

  public static void main(String[] args) {
     Complex n1 = new Complex(2.3, 4.5),
           n2 = new Complex(3.4, 5.0),
           temp;

     temp = add(n1, n2);

     System.out.printf("Sum = %.1f + %.1fi", temp.real, temp.imag);
  }

  public static Complex add(Complex n1, Complex n2)
  {
     Complex temp = new Complex(0.0, 0.0);

     temp.real = n1.real + n2.real;
     temp.imag = n1.imag + n2.imag;
```

```
    return(temp);
  }
}
**************************************************************************
Example: Calculate Difference Between Two Time Periods
public class Time {

  int seconds;
  int minutes;
  int hours;

  public Time(int hours, int minutes, int seconds) {
    this.hours = hours;
    this.minutes = minutes;
    this.seconds = seconds;
  }

  public static void main(String[] args) {

   // create objects of Time class
    Time start = new Time(8, 12, 15);
    Time stop = new Time(12, 34, 55);
    Time diff;

    // call difference method
    diff = difference(start, stop);

    System.out.printf("TIME DIFFERENCE: %d:%d:%d - ", start.hours, start.minutes, start.seconds);
    System.out.printf("%d:%d:%d", stop.hours, stop.minutes, stop.seconds);
    System.out.printf("= %d:%d:%d\n", diff.hours, diff.minutes, diff.seconds);
  }

  public static Time difference(Time start, Time stop)
  {
    Time diff = new Time(0, 0, 0);

    // if start second is greater
    // convert minute of stop into seconds
    // and add seconds to stop second
    if(start.seconds > stop.seconds){
      --stop.minutes;
      stop.seconds += 60;
    }

    diff.seconds = stop.seconds - start.seconds;

    // if start minute is greater
    // convert stop hour into minutes
    // and add minutes to stop minutes
    if(start.minutes > stop.minutes){
      --stop.hours;
      stop.minutes += 60;
    }

    diff.minutes = stop.minutes - start.minutes;
```

```
    diff.hours = stop.hours - start.hours;

    // return the difference time
    return(diff);
  }
}
```

****

Example 1: Program to print half pyramid using *

```
*
* *
* * *
* * * *
* * * * *
```

Source code

```
public class Main {

  public static void main(String[] args) {
   int rows = 5;

   for (int i = 1; i <= rows; ++i) {
    for (int j = 1; j <= i; ++j) {
    System.out.print("* ");
    }
    System.out.println();
   }
  }
}
```

****

Example 2: Program to print half pyramid a using numbers

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

Source Code

```
public class Main {

  public static void main(String[] args) {
   int rows = 5;

   for (int i = 1; i <= rows; ++i) {
    for (int j = 1; j <= i; ++j) {
    System.out.print(j + " ");
    }
    System.out.println();
   }
  }
}
```

****

Example 3: Program to print half pyramid using alphabets

```
A B
B
C C C
```

D D D D
E E E E E
Source Code

```
public class Main {

  public static void main(String[] args) {
    char last = 'E', alphabet = 'A';

    for (int i = 1; i <= (last - 'A' + 1); ++i) {
      for ( int j = 1; j <= i; ++j) {
      System.out.print(alphabet + " ");
      }
      ++alphabet;

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

****

Example 4: Inverted half pyramid using *

```
* * * * *
* * * *
* * *
* *
*
```

Source Code

```
public class Main {

  public static void main(String[] args) {
    int rows = 5;

    for (int i = rows; i >= 1; --i) {
      for ( int j = 1; j <= i; ++j) {
       System.out.print("* ");
      }
      System.out.println();
    }
  }
}
```

****

Example 5: Inverted half pyramid using numbers

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

Source Code

```
public class Main {

  public static void main(String[] args) {
    int rows = 5;
```

```
  for (int i = rows; i >= 1; --i) {
    for (int j = 1; j <= i; ++j) {
     System.out.print(j + " ");
    }
    System.out.println();
   }
  }
}
```

****

Example 6: Program to print full pyramid using *

```
     *
    * * *
   * * * * *
  * * * * * * *
* * * * * * * * *
```

Source Code

```
public class Main {

 public static void main(String[] args) {
  int rows = 5, k = 0;

  for (int i = 1; i <= rows; ++i, k = 0) {
   for (int space = 1; space <= rows - i; ++space) {
    System.out.print(" ");
   }

   while (k != 2 * i - 1) {
    System.out.print("* ");
    ++k;
   }

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

****

Example 7: Program to print pyramid using numbers

```
    1
   2 3 2
  3 4 5 4 3
 4 5 6 7 6 5 4
5 6 7 8 9 8 7 6 5
```

Source Code

```
public class Main {

 public static void main(String[] args) {
  int rows = 5, k = 0, count = 0, count1 = 0;

  for (int i = 1; i <= rows; ++i) {
   for (int space = 1; space <= rows - i; ++space) {
    System.out.print(" ");
    ++count;
```

```
    }
    while (k != 2 * i - 1) {
     if (count <= rows - 1) {
      System.out.print((i + k) + " ");
      ++count;
     } else {
      ++count1;
      System.out.print((i + k - 2 * count1) + " ");
     }

     ++k;
    }
    count1 = count = k = 0;

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

****

Example 8: Inverted full pyramid using *

```
* * * * * * * * *
 * * * * * * *
  * * * * *
   * * *
    *
```

Source Code

```
public class Main {

 public static void main(String[] args) {
  int rows = 5;

  for(int i = rows; i >= 1; --i) {
   for(int space = 1; space <= rows - i; ++space) {
    System.out.print("  ");
   }

   for(int j=i; j <= 2 * i - 1; ++j) {
    System.out.print("* ");
   }

   for(int j = 0; j < i - 1; ++j) {
    System.out.print("* ");
   }

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

****

Example 9: Print Pascal's triangle

```
     1
    1 1
```

```
   1  2  1
  1  3  3  1
 1  4  6  4  1
1  5  10  10 5  1
```

Source Code

```
public class Main {

 public static void main(String[] args) {
  int rows = 6, coef = 1;

  for(int i = 0; i < rows; i++) {
   for(int space = 1; space < rows - i; ++space) {
    System.out.print("  ");
   }

   for(int j = 0; j <= i; j++) {
    if (j == 0 || i == 0)
     coef = 1;
    else
     coef = coef * (i - j + 1) / j;

    System.out.printf("%4d", coef);
   }
   System.out.println();
  }
 }
}
```

****

Example 10: Print Floyd's Triangle.

```
1
2 3
4 5 6
7 8 9 10
```

Source Code

```
public class Main {

 public static void main(String[] args) {
  int rows = 4, number = 1;

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

   for(int j = 1; j <= i; j++) {
    System.out.print(number + " ");
    ++number;
   }

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

****

Example 1: Program to Remove All Whitespaces

```
public class Whitespaces {

  public static void main(String[] args) {
    String sentence = "T    his is b  ett    er.";
    System.out.println("Original sentence: " + sentence);

    sentence = sentence.replaceAll("\\s", "");
    System.out.println("After replacement: " + sentence);
  }
}
```

****

Example 2: Take string from users and remove the white space

```
import java.util.Scanner;

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

  // create an object of Scanner
  Scanner sc = new Scanner(System.in);
  System.out.println("Enter the string");

  // take the input
  String input = sc.nextLine();
  System.out.println("Original String: " + input);

  // remove white spaces
  input = input.replaceAll("\\s", "");
  System.out.println("Final String: " + input);
  sc.close();
 }
}
```

****

Example 1: Print an Array using For loop

```
public class Array {

  public static void main(String[] args) {
    int[] array = {1, 2, 3, 4, 5};

    for (int element: array) {
      System.out.println(element);
    }
  }
}
```

****

Example 2: Print an Array using standard library Arrays

```
import java.util.Arrays;

public class Array {

  public static void main(String[] args) {
    int[] array = {1, 2, 3, 4, 5};

    System.out.println(Arrays.toString(array));
  }
}
```

```
*********************************************************************************
Example 3: Print a Multi-dimensional Array
import java.util.Arrays;

public class Array {

  public static void main(String[] args) {
    int[][] array = {{1, 2}, {3, 4}, {5, 6, 7}};

    System.out.println(Arrays.deepToString(array));
  }
}
*********************************************************************************
Example 1: Convert String to Date using predefined formatters
import java.time.LocalDate;
import java.time.format.DateTimeFormatter;

public class TimeString {

  public static void main(String[] args) {
    // Format y-M-d or yyyy-MM-d
    String string = "2017-07-25";
    LocalDate date = LocalDate.parse(string, DateTimeFormatter.ISO_DATE);

    System.out.println(date);
  }
}
*********************************************************************************
Example 2: Convert String to Date using pattern formatters
import java.time.LocalDate;
import java.time.format.DateTimeFormatter;
import java.util.Locale;

public class TimeString {

  public static void main(String[] args) {
    String string = "July 25, 2017";

    DateTimeFormatter formatter = DateTimeFormatter.ofPattern("MMMM d, yyyy", Locale.ENGLISH);
    LocalDate date = LocalDate.parse(string, formatter);

    System.out.println(date);
  }
}
*********************************************************************************
Example 1: Round a Number using format
public class Decimal {

  public static void main(String[] args) {
    double num = 1.34567;

    System.out.format("%.4f", num);
  }
}
*********************************************************************************
```

Example 2: Round a Number using DecimalFormat

```
import java.math.RoundingMode;
import java.text.DecimalFormat;

public class Decimal {

  public static void main(String[] args) {
    double num = 1.34567;
    DecimalFormat df = new DecimalFormat("#.###");
    df.setRoundingMode(RoundingMode.CEILING);

    System.out.println(df.format(num));
  }
}
```

****

Example 1: Concatenate Two Arrays using arraycopy

```
import java.util.Arrays;

public class Concat {

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

    int aLen = array1.length;
    int bLen = array2.length;
    int[] result = new int[aLen + bLen];

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

****

Example 2: Concatenate Two Arrays without using arraycopy

```
import java.util.Arrays;

public class Concat {

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

    int length = array1.length + array2.length;

    int[] result = new int[length];
    int pos = 0;
    for (int element : array1) {
      result[pos] = element;
      pos++;
    }

    for (int element : array2) {
      result[pos] = element;
```

```
            pos++;
        }

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

****

Example 1: Convert char to String

```
public class CharString {

    public static void main(String[] args) {
        char ch = 'c';
        String st = Character.toString(ch);
        // Alternatively
        // st = String.valueOf(ch);

        System.out.println("The string is: " + st);
    }
}
```

****

Example 2: Convert char array to String

If you have a char array instead of just a char, we can easily convert it to String using String methods as follows:

```
public class CharString {

    public static void main(String[] args) {
        char[] ch = {'a', 'e', 'i', 'o', 'u'};

        String st = String.valueOf(ch);
        String st2 = new String(ch);

        System.out.println(st);
        System.out.println(st2);
    }
}
```

****

Example 3: Convert String to char array

We can also convert a string to char array (but not char) using String's method toCharArray().

```
import java.util.Arrays;

public class StringChar {

    public static void main(String[] args) {
        String st = "This is great";

        char[] chars = st.toCharArray();
        System.out.println(Arrays.toString(chars));
    }
}
```

****

Example 1: Check if Int Array contains a given value

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

```
  int[] num = {1, 2, 3, 4, 5};
  int toFind = 3; boolean
  found = false;

  for (int n : num) {
   if (n == toFind) {
    found = true;
    break;
   }
  }

  if(found)
   System.out.println(toFind + " is found.");
  else
   System.out.println(toFind + " is not found.");
 }
}
```

****

Example 2: Check if an array contains the given value using Stream

```
import java.util.stream.IntStream;

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

  int[] num = {1, 2, 3, 4, 5};
  int toFind = 7;

  boolean found = IntStream.of(num).anyMatch(n -> n == toFind);

  if(found)
   System.out.println(toFind + " is found.");
  else
   System.out.println(toFind + " is not found.");

 }
}
```

****

Example 3: Check if an array contains a given value for non-primitive types

```
import java.util.Arrays;

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

  String[] strings = {"One","Two","Three","Four","Five"};
  String toFind= "Four";

  boolean found = Arrays.stream(strings).anyMatch(t -> t.equals(toFind));

  if(found)
   System.out.println(toFind + " is found.");
  else
   System.out.println(toFind + " is not found.");
 }
}
```

```
**********************************************************************
Example 1: Check if String is Empty or Null
class Main {
  public static void main(String[] args) {

    // create null, empty, and regular strings
    String str1 = null;
    String str2 = "";
    String str3 = "  ";

    // check if str1 is null or empty
    System.out.println("str1 is " + isNullEmpty(str1));

    // check if str2 is null or empty
    System.out.println("str2 is " + isNullEmpty(str2));

    // check if str3 is null or empty
    System.out.println("str3 is " + isNullEmpty(str3));
  }

  // method check if string is null or empty
  public static String isNullEmpty(String str) {

    // check if string is null
    if (str == null) {
      return "NULL";
    }

    // check if string is empty
    else if(str.isEmpty()){
    return "EMPTY";
    }

    else {
      return "neither NULL nor EMPTY";
    }
  }
}
**********************************************************************
Example 2: Check if String with spaces is Empty or Null
class Main {
  public static void main(String[] args) {

    // create a string with white spaces
    String str = "    ";

    // check if str1 is null or empty
    System.out.println("str is " + isNullEmpty(str));
  }

  // method check if string is null or empty
  public static String isNullEmpty(String str) {

    // check if string is null
    if (str == null) {
```

```
    return "NULL";
   }

   // check if string is empty
   else if(str.trim().isEmpty()){
   return "EMPTY";
   }

   else {
    return "neither NULL nor EMPTY";
   }
  }
}
```

****

Example 1: Get Current date and time in default format

```
import java.time.LocalDateTime;

public class CurrentDateTime {

   public static void main(String[] args) {
      LocalDateTime current = LocalDateTime.now();

      System.out.println("Current Date and Time is: " + current);
   }
}
```

****

Example 2: Get Current date and time with pattern

```
import java.time.LocalDateTime;
import java.time.format.DateTimeFormatter;

public class CurrentDateTime {

   public static void main(String[] args) {
      LocalDateTime current = LocalDateTime.now();

      DateTimeFormatter formatter = DateTimeFormatter.ofPattern("yyyy-MM-dd HH:mm:ss.SSS");
      String formatted = current.format(formatter);

      System.out.println("Current Date and Time is: " + formatted);
   }
}
```

****

Example 3: Get Current Datetime using predefined constants

```
import java.time.LocalDateTime;
import java.time.format.DateTimeFormatter;

public class CurrentDateTime {

   public static void main(String[] args) {
      LocalDateTime current = LocalDateTime.now();

      DateTimeFormatter formatter = DateTimeFormatter.BASIC_ISO_DATE;
      String formatted = current.format(formatter);

      System.out.println("Current Date is: " + formatted);
```

```
  }
}
```

****

Example 4: Get Current Datetime in localized style

```
import java.time.LocalDateTime;
import java.time.format.DateTimeFormatter;
import java.time.format.FormatStyle;

public class CurrentDateTime {

  public static void main(String[] args) {
    LocalDateTime current = LocalDateTime.now();

    DateTimeFormatter formatter = DateTimeFormatter.ofLocalizedDateTime(FormatStyle.MEDIUM);
    String formatted = current.format(formatter);

    System.out.println("Current Date is: " + formatted);
  }
}
```

****

Example 1: Milliseconds to Seconds/Minutes Using toMinutes() and toSeconds()

```
import java.util.concurrent.TimeUnit;

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

  long milliseconds = 1000000;

  // us of toSeconds()
  // to convert milliseconds to minutes
  long seconds = TimeUnit.MILLISECONDS.toSeconds(milliseconds);
  System.out.println(milliseconds + " Milliseconds = " + seconds + " Seconds");

  // use of toMinutes()
  // to convert milliseconds to minutes
  long minutes = TimeUnit.MILLISECONDS.toMinutes(milliseconds);
  System.out.println(milliseconds + " Milliseconds = " + minutes + " Minutes");
 }
}
```

****

Example 2: Milliseconds to Seconds/Minutes Using Mathematical Formula

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

  long milliseconds = 1000000;

  long seconds = (milliseconds / 1000);
  System.out.println(milliseconds + " Milliseconds = " + seconds + " Seconds");

  long minutes = (milliseconds / 1000) / 60;
  System.out.println(milliseconds + " Milliseconds = " + minutes + " Minutes");
 }
}
```

****

Example: Java program to add two dates

```
import java.util.Calendar;

public class AddDates {

   public static void main(String[] args) {

      Calendar c1 = Calendar.getInstance();
      Calendar c2 = Calendar.getInstance();
      Calendar cTotal = (Calendar) c1.clone();

      cTotal.add(Calendar.YEAR, c2.get(Calendar.YEAR));
      cTotal.add(Calendar.MONTH, c2.get(Calendar.MONTH) + 1); // Zero-based months
      cTotal.add(Calendar.DATE, c2.get(Calendar.DATE));
      cTotal.add(Calendar.HOUR_OF_DAY, c2.get(Calendar.HOUR_OF_DAY));
      cTotal.add(Calendar.MINUTE, c2.get(Calendar.MINUTE));
      cTotal.add(Calendar.SECOND, c2.get(Calendar.SECOND));
      cTotal.add(Calendar.MILLISECOND, c2.get(Calendar.MILLISECOND));

      System.out.format("%s + %s = %s", c1.getTime(), c2.getTime(), cTotal.getTime());

   }
}
```

****

Example 1: Join Two Lists using addAll()

```
import java.util.ArrayList;
import java.util.List;

public class JoinLists {

   public static void main(String[] args) {

      List<String> list1 = new ArrayList<String>();
      list1.add("a");

      List<String> list2 = new ArrayList<String>();
      list2.add("b");

      List<String> joined = new ArrayList<String>();

      joined.addAll(list1);
      joined.addAll(list2);

      System.out.println("list1: " + list1);
      System.out.println("list2: " + list2);
      System.out.println("joined: " + joined);

   }
}
```

****

Example 2: Join Two Lists using union()

```
import java.util.ArrayList;
import java.util.List;
import org.apache.commons.collections.ListUtils;

public class JoinLists {
```

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

    List<String> list1 = new ArrayList<String>();
    list1.add("a");

    List<String> list2 = new ArrayList<String>();
    list2.add("b");

    List<String> joined = ListUtils.union(list1, list2);

    System.out.println("list1: " + list1);
    System.out.println("list2: " + list2);
    System.out.println("joined: " + joined);

  }
}
```

****

Example 3: Join Two Lists using stream

```
import java.util.ArrayList;
import java.util.List;
import java.util.stream.Collectors;
import java.util.stream.Stream;

public class JoinLists {

  public static void main(String[] args) {

    List<String> list1 = new ArrayList<String>();
    list1.add("a");

    List<String> list2 = new ArrayList<String>();
    list2.add("b");

    List<String> joined = Stream.concat(list1.stream(), list2.stream())
        .collect(Collectors.toList());

    System.out.println("list1: " + list1);
    System.out.println("list2: " + list2);
    System.out.println("joined: " + joined);

  }
}
```

****

Example 1: Convert the Arraylist into Array

```
import java.util.ArrayList;

class Main {
  public static void main(String[] args) {
    ArrayList<String> languages = new ArrayList<>();

    // Add elements in the arraylist
    languages.add("Java");
    languages.add("Python");
    languages.add("JavaScript");
```

```
    System.out.println("ArrayList: " + languages);

    // Create a new array of String type
    String[] arr = new String[languages.size()];

    // Convert ArrayList into the string array
    languages.toArray(arr);
    System.out.print("Array: ");
    for(String item:arr) {
      System.out.print(item+", ");
    }
  }
}
```

****

Example 2: Convert Array to ArrayList

```
import java.util.Arrays;
import java.util.ArrayList;

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

    // create an array
    String[] array = {"Java", "Python", "C"};
    System.out.println("Array: " + Arrays.toString(array));

    // convert array to arraylist
    ArrayList<String> languages = new ArrayList<>(Arrays.asList(array));

    System.out.println("ArrayList: " + languages);

  }
}
```

****

Example 1: Get current working directory

```
public class CurrDirectory {

  public static void main(String[] args) {

    String path = System.getProperty("user.dir");

    System.out.println("Working Directory = " + path);

  }
}
```

****

Example 2: Get the current working directory using Path

```
import java.nio.file.Paths;

public class CurrDirectory {

  public static void main(String[] args) {

    String path = Paths.get("").toAbsolutePath().toString();
    System.out.println("Working Directory = " + path);
```

```
    }
}
*****************************************************************************
Example 1: Convert Map to List
import java.util.*;

public class MapList {

    public static void main(String[] args) {

        Map<Integer, String> map = new HashMap<>();
        map.put(1, "a");
        map.put(2, "b");
        map.put(3, "c");
        map.put(4, "d");
        map.put(5, "e");

        List<Integer> keyList = new ArrayList(map.keySet());
        List<String> valueList = new ArrayList(map.values());

        System.out.println("Key List: " + keyList);
        System.out.println("Value List: " + valueList);

    }
}
*****************************************************************************
Example 2: Convert Map to List using stream
import java.util.*;
import java.util.stream.Collectors;

public class MapList {

    public static void main(String[] args) {

        Map<Integer, String> map = new HashMap<>();
        map.put(1, "a");
        map.put(2, "b");
        map.put(3, "c");
        map.put(4, "d");
        map.put(5, "e");

        List<Integer> keyList = map.keySet().stream().collect(Collectors.toList());
        List<String> valueList = map.values().stream().collect(Collectors.toList());

        System.out.println("Key List: " + keyList);
        System.out.println("Value List: " + valueList);

    }
}
*****************************************************************************
Example 1: Convert Array to Set
import java.util.*;

public class ArraySet {
```

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

        String[] array = {"a", "b", "c"};
        Set<String> set = new HashSet<>(Arrays.asList(array));

        System.out.println("Set: " + set);
    }
}
```

****

Example 2: Convert Array to Set using stream

```
import java.util.*;

public class ArraySet {

    public static void main(String[] args) {

        String[] array = {"a", "b", "c"};
        Set<String> set = new HashSet<>(Arrays.stream(array).collect(Collectors.toSet()));

        System.out.println("Set: " + set);

    }
}
```

****

Example 3: Convert Set to Array

```
import java.util.*;

public class SetArray {

    public static void main(String[] args) {

        Set<String> set = new HashSet<>();
        set.add("a");
        set.add("b");
        set.add("c");

        String[] array = new String[set.size()];
        set.toArray(array);

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

    }
}
```

****

Example 1: Convert Byte Array to Hex value

```
public class ByteHex {

    public static void main(String[] args) {

        byte[] bytes = {10, 2, 15, 11};

        for (byte b : bytes) {
            String st = String.format("%02X", b);
            System.out.print(st);
        }
```

```
    }
}
**********************************************************************************
Example 2: Convert Byte Array to Hex value using byte operations
public class ByteHex {

  private final static char[] hexArray = "0123456789ABCDEF".toCharArray();
  public static String bytesToHex(byte[] bytes) {
    char[] hexChars = new char[bytes.length * 2];
    for ( int j = 0; j < bytes.length; j++ ) {
      int v = bytes[j] & 0xFF;
      hexChars[j * 2] = hexArray[v >>> 4];
      hexChars[j * 2 + 1] = hexArray[v & 0x0F];
    }
    return new String(hexChars);
  }

  public static void main(String[] args) {

    byte[] bytes = {10, 2, 15, 11};

    String s = bytesToHex(bytes);
    System.out.println(s);

  }
}
**********************************************************************************
Example 1: Create String from file
import java.io.IOException;
import java.nio.charset.Charset;
import java.nio.charset.StandardCharsets;
import java.nio.file.Files;
import java.nio.file.Paths;
import java.util.List;

public class FileString {

  public static void main(String[] args) throws IOException {

    String path = System.getProperty("user.dir") + "\\src\\test.txt";
    Charset encoding = Charset.defaultCharset();

    List<String> lines = Files.readAllLines(Paths.get(path), encoding);
    System.out.println(lines);
  }
}
**********************************************************************
Example 2: Create String from a file
import java.io.IOException;
import java.nio.charset.Charset;
import java.nio.file.Files;
import java.nio.file.Paths;

public class FileString {
```

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

        String path = System.getProperty("user.dir") + "\\src\\test.txt";
        Charset encoding = Charset.defaultCharset();

        byte[] encoded = Files.readAllBytes(Paths.get(path));
        String lines = new String(encoded, encoding);
        System.out.println(lines);
    }
}
```

****

Example 1: Append text to existing file

```
import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Paths;
import java.nio.file.StandardOpenOption;

public class AppendFile {

    public static void main(String[] args) {

        String path = System.getProperty("user.dir") + "\\src\\test.txt";
        String text = "Added text";

        try {
            Files.write(Paths.get(path), text.getBytes(), StandardOpenOption.APPEND);
        } catch (IOException e) {
        }
    }
}
```

****

Example 2: Append text to an existing file using FileWriter

```
import java.io.FileWriter;
import java.io.IOException;

public class AppendFile {

    public static void main(String[] args) {

        String path = System.getProperty("user.dir") + "\\src\\test.txt";
        String text = "Added text";

        try {
            FileWriter fw = new FileWriter(path, true);
            fw.write(text);
            fw.close();
        }
        catch(IOException e) {
        }
    }
}
```

****

Example: Convert stack trace to a string

```
import java.io.PrintWriter;
import java.io.StringWriter;
```

```
public class PrintStackTrace {

   public static void main(String[] args) {

      try {
         int division = 0 / 0;
      } catch (ArithmeticException e) {
         StringWriter sw = new StringWriter();
         e.printStackTrace(new  PrintWriter(sw));
         String exceptionAsString  = sw.toString();
         System.out.println(exceptionAsString);
      }
   }
}
```

****

Example 1: Convert File to byte[]

```
import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Paths;
import java.util.Arrays;

public class FileByte {

   public static void main(String[] args) {

      String path = System.getProperty("user.dir") + "\\src\\test.txt";

      try {
         byte[] encoded = Files.readAllBytes(Paths.get(path));
         System.out.println(Arrays.toString(encoded));
      } catch (IOException e) {

      }
   }
}
```

****

Example 2: Convert byte[] to File

```
import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Paths;

public class ByteFile {

   public static void main(String[] args) {

      String path = System.getProperty("user.dir") + "\\src\\test.txt";
      String finalPath = System.getProperty("user.dir") + "\\src\\final.txt";

      try {
         byte[] encoded = Files.readAllBytes(Paths.get(path));
         Files.write(Paths.get(finalPath), encoded);
      } catch (IOException e) {

      }
```

```
  }
}
*****************************************************************************
Example: Convert InputStream to String
import java.io.*;

public class InputStreamString {

  public static void main(String[] args) throws IOException {

    InputStream stream = new ByteArrayInputStream("Hello there!".getBytes());
    StringBuilder sb = new StringBuilder();
    String line;

    BufferedReader br = new BufferedReader(new InputStreamReader(stream));
    while ((line = br.readLine()) != null) {
      sb.append(line);
    }
    br.close();

    System.out.println(sb);

  }
}
*****************************************************************************
Example: Convert OutputStream to String
import java.io.*;

public class OutputStreamString {

  public static void main(String[] args) throws IOException {

    ByteArrayOutputStream stream = new ByteArrayOutputStream();
    String line = "Hello there!";

    stream.write(line.getBytes());
    String finalString = new String(stream.toByteArray());

    System.out.println(finalString);
  }
}
*****************************************************************************
Example: Lookup enum by string value
public class EnumString {

  public enum TextStyle {
    BOLD, ITALICS, UNDERLINE, STRIKETHROUGH
  }

  public static void main(String[] args) {

    String style = "Bold";

    TextStyle textStyle = TextStyle.valueOf(style.toUpperCase());
```

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

****

Example 1: Compare two strings

```
public class CompareStrings {

  public static void main(String[] args) {

    String style = "Bold";
    String style2 = "Bold";

    if(style == style2)
      System.out.println("Equal");
    else
      System.out.println("Not Equal");
  }
}
```

****

Example 2: Compare two strings using equals()

```
public class CompareStrings {

  public static void main(String[] args) {

    String style = new String("Bold");
    String style2 = new String("Bold");

    if(style.equals(style2))
      System.out.println("Equal");
    else
      System.out.println("Not Equal");
  }
}
```

****

Example 4: Different ways to compare two strings

Here is the string comparison which is possible in Java.

```
public class CompareStrings {

  public static void main(String[] args) {

    String style = new String("Bold");
    String style2 = new String("Bold");

    boolean result = style.equals("Bold"); // true
    System.out.println(result);

    result = style2 == "Bold"; // false
    System.out.println(result);

    result = style == style2; // false
    System.out.println(result);

    result = "Bold" == "Bold"; // true
    System.out.println(result);
```

```
  }
}
```

****

Example: Sort a map by values

```
import java.util.*;

public class Main {

 public static void main(String[] args) {

  // create a map and store elements to it
  LinkedHashMap capitals = new LinkedHashMap();
  capitals.put("Nepal", "Kathmandu");
  capitals.put("India", "New Delhi");
  capitals.put("United States", "Washington");
  capitals.put("England", "London");
  capitals.put("Australia", "Canberra");

  // call the sortMap() method to sort the map
  Map result = sortMap(capitals);

  for (Map.Entry entry : result.entrySet()) {
   System.out.print("Key: " + entry.getKey());
   System.out.println(" Value: " + entry.getValue());
  }
 }

 public static LinkedHashMap sortMap(LinkedHashMap map) {
  List> capitalList = new LinkedList(map.entrySet());

  // call the sort() method of Collections
  Collections.sort(capitalList, (l1, l2) -> l1.getValue().compareTo(l2.getValue()));

  // create a new map
  LinkedHashMap result = new LinkedHashMap();

  // get entry from list to the map
  for (Map.Entry entry : capitalList) {
   result.put(entry.getKey(), entry.getValue());
  }

  return result;
 }
}
```

****

Example: Sort ArrayList of Custom Objects By Property

```
import java.util.*;

public class CustomObject {

  private String customProperty;

  public CustomObject(String property) {
    this.customProperty = property;
  }
```

```
  public String getCustomProperty() {
    return this.customProperty;
  }

  public static void main(String[] args) {

    ArrayList<Customobject> list = new ArrayList<>();
    list.add(new CustomObject("Z"));
    list.add(new    CustomObject("A"));
    list.add(new    CustomObject("B"));
    list.add(new    CustomObject("X"));
    list.add(new CustomObject("Aa"));

    list.sort((o1, o2) -> o1.getCustomProperty().compareTo(o2.getCustomProperty()));

    for (CustomObject obj : list) {
      System.out.println(obj.getCustomProperty());
    }
  }
}
```

```
*********************************************************************************
```

Example 1: Check if a string is numeric

```
public class Numeric {

  public static void main(String[] args) {

    String string = "12345.15";
    boolean numeric = true;

    try {
      Double num = Double.parseDouble(string);
    } catch (NumberFormatException e) {
      numeric = false;
    }

    if(numeric)
      System.out.println(string + " is a number");
    else
      System.out.println(string + " is not a number");
  }
}
```

```
*********************************************************************************
```

Example 1: Create a new directory in Java

```
import java.io.File;

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

  // creates a file object with specified path
  File file = new File("Java Example\\directory");

  // tries to create a new directory
  boolean value = file.mkdir();
  if(value) {
```

```
    System.out.println("The new directory is created.");
   }
   else {
    System.out.println("The directory already exists.");
   }
  }
}
```

****

Example 2: Create a new Directory using the mkdirs() method

```
import java.io.File;

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

    // creates a file object in the current path
    File file = new File("Java Tutorial\\directory");

    // tries to create a new directory
    boolean value = file.mkdirs();
    if(value) {
      System.out.println("The new directory is created.");
    }
    else {
      System.out.println("The directory already exists.");
    }
  }
}
```

****

Example: Rename a File in Java

```
import java.io.File;

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

    // create a file object
    File file = new File("oldName");

    // create a file
    try {
      file.createNewFile();
    }
    catch(Exception e) {
      e.getStackTrace();
    }

    // create an object that contains the new name of file
    File newFile = new File("newName");

    // change the name of file
    boolean value = file.renameTo(newFile);

    if(value) {
      System.out.println("The name of the file is changed.");
    }
    else {
```

```
    System.out.println("The name cannot be changed.");
   }
  }
}
```

****

Example 1: Java Program to List all files

```
import java.io.File;

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

    // creates a file object
    File file = new File("C:\\Users\\Guest User\\Desktop\\Java File\\List Method");

    // returns an array of all files
    String[] fileList = file.list();

    for(String str : fileList) {
      System.out.println(str);
    }
  }
}
```

****

Example: Copy files using i/o streams

```
import java.io.FileInputStream;
import java.io.FileOutputStream;

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

    byte[] array = new byte[50];
    try {
      FileInputStream sourceFile = new FileInputStream("input.txt");
      FileOutputStream destFile = new FileOutputStream("newFile");

      // reads all data from input.txt
      sourceFile.read(array);

      // writes all data to newFile
      destFile.write(array);
      System.out.println("The input.txt file is copied to newFile.");

      // closes the stream
      sourceFile.close();
      destFile.close();
    }
    catch (Exception e) {
      e.getStackTrace();
    }
  }
}
```

****

Example 1: Java Program to Convert char to int

```
class Main {
```

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

    // create char variables
    char a = '5';
    char b = 'c';

    // convert char variables to int
    // ASCII value of characters is assigned
    int num1 = a;
    int num2 = b;

    // print the values
    System.out.println(num1);   // 53
    System.out.println(num2);   // 99
  }
}
```

****

Example 2: char to int using getNumericValue() method

We can also use the getNumericValue() method of the Character class to convert the char type variable into int type.

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

    // create char variables
    char a = '5';
    char b = '9';

    // convert char variables to int
    // Use getNumericValue()
    int num1 = Character.getNumericValue(a);
    int num2 = Character.getNumericValue(b);

    // print the numeric value of characters
    System.out.println(num1);   // 5
    System.out.println(num2);   // 9
  }
}
```

****

Example 3: char to int using parseInt() method

We can also use the parseInt() method of the Integer class to convert the char type variable to an int type.

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

    // create char variables
    char a = '5';
    char b = '9';

    // convert char variables to int
    // Use parseInt()
    int num1 = Integer.parseInt(String.valueOf(a));
    int num2 = Integer.parseInt(String.valueOf(b));
```

```
    // print numeric value
    System.out.println(num1);   // 5
    System.out.println(num2);   // 9
  }
}
```

****

Example 4: char to int by subtracting with '0'

In Java, we can also convert the character into an integer by subtracting it with character 0. For example,

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

    // create char variables
    char a = '9';
    char b = '3';

    // convert char variables to int
    // by subtracting with char 0
    int num1 = a - '0';
    int num2 = b - '0';

    // print numeric value
    System.out.println(num1);   // 9
    System.out.println(num2);   // 3
  }
}
```

****

Example 1: Java Program to Convert int to char

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

    // create int variables
    int num1 = 80;
    int num2 = 81;

    // convert int to char
    // typecasting
    char a = (char)num1;
    char b = (char)num2;

    // print value
    System.out.println(a);   // P
    System.out.println(b);   // Q
  }
}
```

****

Example 2: int to char by using forDigit()

We can also use the forDigit() method of the Character class to convert the int type variable into char type.

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

    // create int variables
    int num1 = 1;
    int num2 = 13;
```

```
    // convert int to char
    // for value between 0-9
    char a = Character.forDigit(num1, 10);

    // for value between 0-9
    char b = Character.forDigit(num2, 16);

    // print value
    System.out.println(a);   // 1
    System.out.println(b);   // d
  }
}
```

****

Example 3: int to char by adding '0'

In Java, we can also convert the integer into a character by adding the character '0' with it. For example,

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

    // create int variables
    int num1 = 1;
    int num2 = 9;

    // convert int to char
    char a = (char)(num1 + '0');
    char b = (char)(num2 + '0');

    // print value
    System.out.println(a);   // 1
    System.out.println(b);   // 9
  }
}
```

****

Example 1: Java Program to Convert long to int using Typecasting

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

    // create long variables
    long a = 2322331L;
    long b = 52341241L;

    // convert long into int
    // using typecasting
    int c = (int)a;
    int d = (int)b;

    System.out.println(c);   // 2322331
    System.out.println(d);   // 52341241
  }
}
```

****

Example 2: long to int conversion using toIntExact()

We can also use the toIntExact() method of the Math class to convert the long value into an int.

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

    // create long variable
    long value1 = 52336L;
    long value2 = -445636L;

    // change long to int
    int num1 = Math.toIntExact(value1);
    int num2 = Math.toIntExact(value2);

    // print the int value
    System.out.println(num1); // 52336
    System.out.println(num2); // -445636
  }
}
```

****

Example 3: Convert object of the Long class to int

In Java, we can also convert the object of wrapper class Long into an int. For this, we can use the intValue() method. For, example,

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

    // create an object of Long class
    Long obj = 52341241L;

    // convert object of Long into int
    // using intValue() int
    a = obj.intValue();

    System.out.println(a);   // 52341241
  }
}
```

****

Example 1: Java Program to Convert int to long using Typecasting

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

    // create int variables
    int a = 25;
    int b = 34;

    // convert int into long
    // using typecasting
    long c = a;
    long d = b;

    System.out.println(c);   // 25
    System.out.println(d);   // 34
  }
}
```

****

Example 2: Java Program to Convert int into object of Long using valueof()

We can convert the int type variable into an object of the Long class. For example,

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

    // create int variables
    int a = 251;

    // convert to an object of Long
    // using valueOf()
    Long obj = Long.valueOf(a);

    System.out.println(obj);   // 251
  }
}
```

****

Example 1: Convert boolean to string using valueOf()

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

    // create boolean variables
    boolean booleanValue1 = true;
    boolean booleanValue2 = false;

    // convert boolean to string
    // using valueOf()
    String stringValue1 = String.valueOf(booleanValue1);
    String stringValue2 = String.valueOf(booleanValue2);

    System.out.println(stringValue1);   // true
    System.out.println(stringValue2);   // true
  }
}
```

****

Example 2: Convert boolean to String using toString()

We can also convert the boolean variables into strings using the toString() method of the Boolean class. For example,

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

    // create boolean variables
    boolean booleanValue1 = true;
    boolean booleanValue2 = false;

    // convert boolean to string
    // using toString()
    String stringValue1 = Boolean.toString(booleanValue1);
    String stringValue2 = Boolean.toString(booleanValue2);

    System.out.println(stringValue1);   // true
    System.out.println(stringValue2);   // true
  }
}
```

****

Example 1: Convert string to boolean using parseBoolean()

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

    // create string variables
    String str1 = "true";
    String str2 = "false";

    // convert string to boolean
    // using parseBoolean()
    boolean b1 = Boolean.parseBoolean(str1);
    boolean b2 = Boolean.parseBoolean(str2);

    // print boolean values
    System.out.println(b1);   // true
    System.out.println(b2);   // false
  }
}
```

****

Example 2: Convert string to boolean using valueOf()

We can also convert the string variables into boolean using the valueOf() method. For example,

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

    // create string variables
    String str1 = "true";
    String str2 = "false";

    // convert string to boolean
    // using valueOf()
    boolean b1 = Boolean.valueOf(str1);
    boolean b2 = Boolean.valueOf(str2);

    // print boolean values
    System.out.println(b1);   // true
    System.out.println(b2);   // false
  }
}
```

****

Example 1: Java Program to Convert string to int using parseInt()

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

    // create string variables
    String str1 = "23";
    String str2 = "4566";

    // convert string to int
    // using parseInt()
    int num1 = Integer.parseInt(str1);
    int num2 = Integer.parseInt(str2);

    // print int values
    System.out.println(num1);   // 23
    System.out.println(num2);   // 4566
```

```
  }
}
**********************************************************************
class Main {
  public static void main(String[] args) {

    // create int variable
    int num1 = 36;
    int num2 = 99;

    // convert int to string
    // using valueOf()
    String str1 = String.valueOf(num1);
    String str2 = String.valueOf(num2);

    // print string variables
    System.out.println(str1);   // 36
    System.out.println(str2);   // 99
  }
}
**********************************************************************************
Example 1: Java Program to Convert int to double using Typecasting
class Main {
  public static void main(String[] args) {

    // create int variables
    int a =33;
    int b = 29;

    // convert int into double
    // using typecasting
    double c = a;
    double d = b;

    System.out.println(c);   // 33.0
    System.out.println(d);   // 29.0
  }
}
**********************************************************************************
Example 1: Java Program to Convert double to int using Typecasting
class Main {
  public static void main(String[] args) {

    // create double variables
    double a = 23.78D;
    double b = 52.11D;

    // convert double into int
    // using typecasting
    int c = (int)a;
    int d = (int)b;

    System.out.println(c);   // 23
    System.out.println(d);   // 52
  }
```

```
}
**************************************************************************
Example 1: Java Program to Convert string to double using parseDouble()
class Main {
  public static void main(String[] args) {

    // create string variables
    String str1 = "23";
    String str2 = "456.6";

    // convert string to double
    // using parseDouble()
    double num1 = Double.parseDouble(str1);
    double num2 = Double.parseDouble(str2);

    // print double values
    System.out.println(num1);   // 23.0
    System.out.println(num2);   // 456.6
  }
}
**********************************************************
Example 1: Java Program to Convert double to string using valueOf()
class Main {
  public static void main(String[] args) {

    // create double variable
    double num1 = 36.33;
    double num2 = 99.99;

    // convert double to string
    // using valueOf()
    String str1 = String.valueOf(num1);
    String str2 = String.valueOf(num2);

    // print string variables
    System.out.println(str1);   // 36.33
    System.out.println(str2);   // 99.99
  }
}
**************************************************************************
Example 1: Java Program to Convert Primitive Types to Wrapper Objects
class Main {
  public static void main(String[] args) {

    // create primitive types
    int var1 = 5;
    double var2 = 5.65;
    boolean var3 = true;

    //converts into wrapper objects
    Integer obj1 = Integer.valueOf(var1);
    Double obj2 = Double.valueOf(var2);
    Boolean obj3 = Boolean.valueOf(var3);

    // checks if obj are objects of
```

```
    // corresponding wrapper class
    if(obj1 instanceof Integer) {
      System.out.println("An object of Integer is created.");
    }

    if(obj2 instanceof Double) {
      System.out.println("An object of Double is created.");
    }

    if(obj3 instanceof Boolean) {
      System.out.println("An object of Boolean is created");
    }
  }
}
```

****

Example: Java program to implement Bubble Sort Algo.

```
// import the Class
import java.util.Arrays;
import java.util.Scanner;

class Main {

  // create an object of scanner
  // to take input from the user
  Scanner input = new Scanner(System.in);

  // method to perform bubble sort
  void bubbleSort(int array[]) {
    int size = array.length;

    // for ascending or descending sort
    System.out.println("Choose Sorting Order:");
    System.out.println("1 for Ascending \n2 for Descending");
    int sortOrder = input.nextInt();

    // run loops two times
    // first loop access each element of the array
    for (int i = 0; i < size - 1; i++)

      // second loop performs the comparison in each iteration
      for (int j = 0; j < size - i - 1; j++)

        // sort the array in ascending order
        if (sortOrder == 1) {
          // compares the adjacent element
          if (array[j] > array[j + 1]) {

            // swap if left element is greater than right
            int temp = array[j];
            array[j] = array[j + 1];
            array[j + 1] = temp;
          }
        }

        // sort the array in descending order
```

```
      else {
       // compares the adjacent element
       if (array[j] < array[j + 1]) {

        // swap if left element is smaller than right
        int temp = array[j];
        array[j] = array[j + 1];
        array[j + 1] = temp;
       }
      }

  }
  // driver code
  public static void main(String args[]) {

   // create an array
   int[] data = { -2, 45,  0, 11, -9 };

   // create an object of Main class
   Main bs = new Main();

   // call the method bubbleSort using object bs
   // pass the array as the method argument
   bs.bubbleSort(data);
   System.out.println("Sorted Array in Ascending Order:");

   // call toString() of Arrays class
   // to convert data into the string
   System.out.println(Arrays.toString(data));
  }
}
```

****

Example: Java Program to Implement Quick Sort Algorithm

```
import java.util.Arrays;


class Quicksort {

 // method to find the partition position

 static int partition(int array[], int low, int high) {


  // choose the rightmost element as pivot

  int pivot = array[high];


  // pointer for greater element
```

```
  int i = (low - 1);

  // traverse through all elements
  // compare each element with pivot
  for (int j = low; j < high; j++) {
   if (array[j] <= pivot) {

    // if element smaller than pivot is found
    // swap it with the greater element pointed by i
    i++;

    // swapping element at i with element at j
    int temp = array[i];
    array[i] = array[j];
    array[j] = temp;
   }

  }

  // swapt the pivot element with the greater element specified by i
  int temp = array[i + 1];
  array[i + 1] = array[high];
  array[high] = temp;

  // return the position from where partition is done
  return (i + 1);
}

static void quickSort(int array[], int low, int high) {
  if (low < high) {

   // find pivot element such that
   // elements smaller than pivot are on the left
```

```
    // elements greater than pivot are on the right
    int pi = partition(array, low, high);

    // recursive call on the left of pivot
    quickSort(array, low, pi - 1);

    // recursive call on the right of pivot
    quickSort(array, pi + 1, high);
   }
  }
}

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

   int[] data = { 8, 7, 2, 1, 0, 9, 6 };
   System.out.println("Unsorted Array");
   System.out.println(Arrays.toString(data));

   int size = data.length;

   // call quicksort() on array data
   Quicksort.quickSort(data, 0, size - 1);

   System.out.println("Sorted Array in Ascending Order ");
   System.out.println(Arrays.toString(data));
  }
}
```

****

Example: Java Program to Implement Merge Sort Algorithm

```
import java.util.Arrays;
```

```
// Merge sort in Java


class Main {

 // Merge two sub arrays L and M into array
 void merge(int array[], int p, int q, int r) {

  int n1 = q - p + 1;

  int n2 = r - q;


  int L[] = new int[n1];

  int M[] = new int[n2];


  // fill the left and right array
  for (int i = 0; i < n1; i++)

   L[i] = array[p + i];
  for (int j = 0; j < n2; j++)
   M[j] = array[q + 1 + j];


  // Maintain current index of sub-arrays and main array
  int i, j, k;

  i = 0;

  j = 0;

  k = p;


  // Until we reach either end of either L or M, pick larger among
  // elements L and M and place them in the correct position at A[p..r]
  // for sorting in descending
  // use if(L[i] >= <[j])
  while (i < n1 && j < n2) {
   if (L[i] <= M[j]) {
    array[k] = L[i];
```

```
    i++;
   } else {
    array[k] = M[j];
    j++;
   }
   k++;
  }

  // When we run out of elements in either L or M,
  // pick up the remaining elements and put in A[p..r]
  while (i < n1) {
   array[k] = L[i];
   i++;
   k++;
  }

  while (j < n2) {
   array[k] = M[j];
   j++;
   k++;
  }
}

// Divide the array into two sub arrays, sort them and merge them
void mergeSort(int array[], int left, int right) {
 if (left < right) {

  // m is the point where the array is divided into two sub arrays
  int mid = (left + right) / 2;

  // recursive call to each sub arrays
  mergeSort(array, left, mid);
  mergeSort(array, mid + 1, right);
```

```
    // Merge the sorted sub arrays
    merge(array, left, mid, right);
  }
 }

 public static void main(String args[]) {

  // created an unsorted array
  int[] array = { 6, 5, 12,  10, 9, 1 };

  Main ob = new Main();

  // call the method mergeSort()
  // pass argument: array, first index and last index
  ob.mergeSort(array, 0, array.length - 1);

  System.out.println("Sorted Array:");
  System.out.println(Arrays.toString(array));
 }
}
```

****

Example: Java Program to Implement Binary Search Algorithm

```
import java.util.Scanner;

// Binary Search in Java

class Main {

 int binarySearch(int array[], int element, int low, int high) {

  // Repeat until the pointers low and high meet each other
  while (low <= high) {
```

```
    // get index of mid element
    int mid = low + (high - low) / 2;

    // if element to be searched is the mid element
    if (array[mid] == element)
      return mid;

    // if element is less than mid element
    // search only the left side of mid
    if (array[mid] < element)
      low = mid + 1;

    // if element is greater than mid element
    // search only the right side of mid
    else
      high = mid - 1;
  }

  return -1;
}

public static void main(String args[]) {

  // create an object of Main class
  Main obj = new Main();

  // create a sorted array
  int[] array = { 3, 4, 5, 6, 7, 8, 9 };
  int n = array.length;

  // get input from user for element to be searched
  Scanner input = new Scanner(System.in);
```

```
    System.out.println("Enter element to be searched:");

    // element to be searched
    int element = input.nextInt();
    input.close();


    // call the binary search method
    // pass arguments: array, element, index of first and last element
    int result = obj.binarySearch(array, element, 0, n - 1);
    if (result == -1)
    System.out.println("Not found");
    else
      System.out.println("Element found at index " + result);
  }
}
```

****

Example 1: Java program to call one constructor from another

```
class Main {

  int sum;

  // first constructor
  Main() {
    // calling the second constructor
    this(5, 2);
  }

  // second constructor
  Main(int arg1, int arg2) {
    // add two value
    this.sum = arg1 + arg2;
  }
```

```
  void display() {
    System.out.println("Sum is: " + sum);
  }

  // main class
  public static void main(String[] args) {

    // call the first constructor
    Main obj = new Main();

    // call display method
    obj.display();
  }
}
```

****

Example 1: Java program to create a private constructor

```
class Test {

  // create private constructor
  private Test () {
    System.out.println("This is a private constructor.");
  }

  // create a public static method
  public static void instanceMethod() {

    // create an instance of Test class
    Test obj = new Test();
  }

}

class Main {
```

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

    // call the instanceMethod()
    Test.instanceMethod();
  }
}
```

****

Example 1: Define lambda expressions as method parameters

```
import java.util.ArrayList;

class Main {
  public static void main(String[] args) {
    // create an ArrayList
    ArrayList<String> languages = new ArrayList<>();

    // add elements to the ArrayList
    languages.add("java");
    languages.add("swift");
    languages.add("python");
    System.out.println("ArrayList: " + languages);

    // pass lambda expression as parameter to replaceAll() method
    languages.replaceAll(e -> e.toUpperCase());
    System.out.println("Updated ArrayList: " + languages);
  }
}
```

****

Example: Java Program to Pass Method as Argument

```
class Main {

  // calculate the sum
  public int add(int a, int b) {
```

```
  // calculate sum
  int sum = a + b;
  return sum;
 }

 // calculate the square public
 void square(int num) { int
 result = num * num;
  System.out.println(result);   // prints 576
 }
 public static void main(String[] args) {

  Main obj = new Main();

  // call the square() method
  // passing add() as an argument
  obj.square(obj.add(15, 9));

 }
}
```

****

Example 1: Java Program to calculate the method execution time

```
class Main {

 // create a method
 public void display() {
  System.out.println("Calculating Method execution time:");
 }

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