

## 1. Java program to Find Odd or Even number

```
import java.util.Scanner;

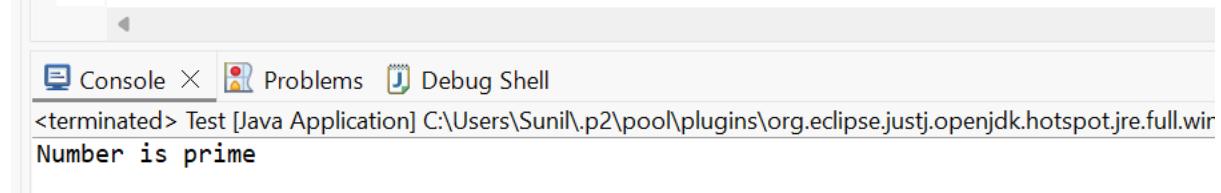
public class OddEven {
    public static void main(String[] args) {

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

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

## 2. Java program to find Prime number

```
3 public class Test {
4     public static void main(String[] args) {
5         int n = 5;
6         for (int i = 2; i <= n / 2; i++) {
7             if (n % i == 0) {
8                 System.out.println("Number is not prime");
9             } else {
10                 System.out.println("Number is prime");
11             }
12         }
13     }
14 }
15
```



### 3. Java program to find Fibonacci series up to a given number range

```
2
3 public class Test {
4     public static void main(String[] args) {
5         int number = 5;
6         int first = 0, second = 1, next;
7         System.out.println("Fibonacci series is ");
8         for ( int i = 0; i<=number; i++)
9             {
10                 System.out.println(first + " ");
11                 next = second+first;
12                 first = second;
13                 second = next;
14             }
15     }
16 }
```

```
<terminated> Test Java Application C:\Users\Sunil\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_21.0
Fibonacci series is
0
1
1
2
3
5
```

### 4. Java program to swap two numbers without using third variable

```
3 public class Test {
4     public static void main(String[] args) {
5         int a = 10;
6         int b = 20;
7
8         System.out.println("Before swapping: a = " + a + ", b = " + b);
9
10        a = a + b; // a now holds the sum of original a and b
11        b = a - b; // b now holds the original value of a
12        a = a - b; // a now holds the original value of b
13
14        System.out.println("After swapping: a = " + a + ", b = " + b);
15    }
16 }
```

```
Console X Problems Debug Shell
<terminated> Test [Java Application] C:\Users\Sunil\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_21.0
Before swapping: a = 10, b = 20
After swapping: a = 20, b = 10
```

## 5. Java program to Find Factorial on given Number

The screenshot shows the Eclipse IDE interface. The code editor displays a Java class named 'Test' with a main method that calculates the factorial of a given number (5 in this case). The console tab at the bottom shows the execution of the program, prompting for input and displaying the result.

```
3 public class Test {  
4     public static void main(String[] args) {  
5         int factorial = 1;  
6         int number = 5;  
7         System.out.print("Enter the number :\n"+number);  
8         for (int i = 1; i <= number; i++) {  
9             factorial = factorial * i;  
10        }  
11        System.out.println("\nFactorial of the number is :\n" + factorial);  
12    }  
13}  
14
```

Console X Problems Debug Shell  
<terminated> Test [Java Application] C:\Users\Sunil\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86\_64\_21.0.7.v210721-1000  
Enter the number :  
5  
Factorial of the number is :  
120

## 6. Java program to Reverse Number

The screenshot shows the Eclipse IDE interface. The code editor displays a Java class named 'Test' with a main method that reverses the digits of a given number (12345). The console tab at the bottom shows the execution of the program, displaying the reversed number.

```
3 public class Test {  
4     public static void main(String[] args) {  
5         int num = 12345; // The number to be reversed  
6         int reversedNum = 0;  
7         while (num != 0) {  
8             int digit = num % 10; // Get the last digit  
9             reversedNum = reversedNum * 10 + digit; // Add the digit to reversedNum  
10            num /= 10; // Remove the last digit from num  
11        }  
12        System.out.println("Reversed Number: " + reversedNum);  
13    }  
14}  
15
```

Console X Problems Debug Shell  
<terminated> Test [Java Application] C:\Users\Sunil\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86\_64\_21.0.7.v210721-1000  
Reversed Number: 54321

## 7. Java program to find Armstrong Number

The screenshot shows the Eclipse IDE interface with the Java code for finding Armstrong numbers. The code uses a while loop to calculate the sum of cubes of digits until the original number becomes zero. It then checks if the calculated result equals the original number to determine if it's an Armstrong number. The output window shows the execution of the program for the number 371, which is confirmed as an Armstrong number. A separate message box displays the mathematical breakdown of 153 being an Armstrong number.

```
5 public class Test {  
6     public static void main(String[] args) {  
7         int number = 371, originalNumber, remainder, result = 0;  
8  
9         originalNumber = number;  
10    }  
11    while (originalNumber != 0)  
12    {  
13        remainder = originalNumber % 10;  
14        result += Math.pow(remainder, 3);  
15        originalNumber /= 10;  
16    }  
17  
18    if (result == number)  
19        System.out.println(number + " is an Armstrong number.");  
20    else  
21        System.out.println(number + " is not an Armstrong number.");  
22    }  
23 }  
24  
Console X Problems Debug Shell  
<terminated> Test [Java Application] C:\Users\Sunil\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64  
371 is an Armstrong number.  
  
153 = 1*1*1 + 5*5*5 + 3*3*3 // 153 is an Armstrong number.
```

## 8. Java program to find number of digits in given number

The screenshot shows the Eclipse IDE interface with the Java code for counting the number of digits in a given integer. The program handles both positive and negative numbers by first converting them to positive before counting. The output window shows the execution of the program for three numbers (12345, 0, and -987), correctly outputting their respective digit counts (5, 1, and 3).

```
4  
5 public class Test {  
6     public static int countDigitsStringConversion(int number) {  
7         // Handle negative numbers by converting to positive before converting to string  
8         return String.valueOf(Math.abs(number)).length();  
9     }  
10    }  
11    public static void main(String[] args) {  
12        int num1 = 12345;  
13        int num2 = 0;  
14        int num3 = -987;  
15  
16        System.out.println("Number of digits in " + num1 + ": " + countDigitsStringConversion(num1)); // Output: 5  
17        System.out.println("Number of digits in " + num2 + ": " + countDigitsStringConversion(num2)); // Output: 1  
18        System.out.println("Number of digits in " + num3 + ": " + countDigitsStringConversion(num3)); // Output: 3  
19    }  
20 }  
21  
Console X Problems Debug Shell  
<terminated> Test [Java Application] C:\Users\Sunil\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_21.0.7.v20250502-0916\jre\bin\javaw.exe (06-Sept-2025,  
Number of digits in 12345: 5  
Number of digits in 0: 1  
Number of digits in -987: 3
```

## 9. Java program to find Palindrome number

```
3 public class Test {  
4     public static void main(String[] args) {  
5         int number = 121;  
6         if (isPalindrome(number)) {  
7             System.out.println(number + " is a palindrome.");  
8         } else {  
9             System.out.println(number + " is not a palindrome.");  
10        }  
11    }  
12  
13    public static boolean isPalindrome(int num) {  
14        int originalNumber = num;  
15        int reversedNumber = 0;  
16        while (num != 0) {  
17            int digit = num % 10;  
18            reversedNumber = reversedNumber * 10 + digit;  
19            num = num / 10;  
20        }  
21        return originalNumber == reversedNumber;  
22    }  
23}
```

Console X Problems Debug Shell  
<terminated> Test [Java Application] C:\Users\Sunil\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86\_64\_2  
121 is a palindrome.

## 10. Java program to calculate the sum of digits of a number

```
5 public class Test {  
6     public static void main(String[] args) {  
7         int number = 12345;  
8         int sum = 0;  
9         while (number > 0) {  
10            int digit = number % 10; // Extract the last digit  
11            sum = sum + digit; // Add the digit to sum  
12            number = number / 10; // Remove the last digit from number  
13        }  
14        System.out.println("Sum of digits of " + number + " is: " + sum);  
15    }  
16}  
17
```

Console X Problems Debug Shell  
<terminated> Test [Java Application] C:\Users\Sunil\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86\_64  
Sum of digits of 0 is: 15