Batch: D3	**Roll No.: 16010123294**
Experiment / assignment / tutorial No.1	
Grade: AA / AB / BB / BC / CC / CD /DD	
Signature of the Staff In-charge with date	

TITLE : Armstrong number

AIM:
Write a Java program to display armstrong numbers in the given range (Make use of a function).

Variations :

Implementation of Program with One class

Accessibility with static and non-static methods within class and outside class.

Expected OUTCOME of Experiment:

CO1:Apply the features of object oriented programming languages. (C++ and Java)
CO2:Explore arrays, vectors, classes and objects in C++ and Java

---

Books/ Journals/ Websites referred:

1. E. Balagurusamy, “Programming with Java”, McGraw-Hill.
2. E. Balagurusamy, “Object Oriented Programming with C++”, McGraw-Hill.

Pre Lab/ Prior Concepts:

The Scanner class is a class in java.util, which allows the user to read values of various types. There are far more methods in class Scanner than you will need in this course. We only cover a small useful subset, ones that allow us to read in numeric values from

either the keyboard or file without having to convert them from strings and determine if there are more values to be read.

Scanner in = new Scanner(System.in); // System.in is an InputStream
Numeric and String Methods

Method	Returns
int nextInt()	Returns the next token as an int. If the next token is not an integer,InputMismatchException is thrown.
long nextLong()	Returns the next token as a long. If the next token is not an integer,InputMismatchException is thrown.
float nextFloat()	Returns the next token as a float. If the next token is not a float or is out of range, InputMismatchException is thrown.
double nextDouble()	Returns the next token as a long. If the next token is not a float or is out of range, InputMismatchException is thrown.
String next()	Finds and returns the next complete token from this scanner and returns it as a string; a token is usually ended by whitespace such as a blank or line break. If not token exists,NoSuchElementException is thrown.
String nextLine()	Returns the rest of the current line, excluding any line separator at the end.
void close()	Closes the scanner.

The Scanner looks for tokens in the input. A token is a series of characters that ends with what Java calls whitespace. A whitespace character can be a blank, a tab character, a carriage return. Thus, if we read a line that has a series of numbers separated by blanks, the scanner will take each number as a separate token. .
The numeric values may all be on one line with blanks between each value or may be on separate lines. Whitespace characters (blanks or carriage returns) act as

separators. The next method returns the next input value as a string, regardless of what is keyed. For example, given the following code segment and data

- int number = in.nextInt();
- float real = in.nextFloat();
- long number2 = in.nextLong();
- double real2 = in.nextDouble();
- String string = in.next();

Algorithm:

Input the Number:

Prompt the user to enter an integer.

Store the entered integer in a variable num.

Check if the Number is an Armstrong Number:

Define a function isArmstrong that takes an integer num as input.

Inside the function:

Initialize Variables:

Store the original value of num in a variable originalNumber.

Initialize a variable sum to 0. This will hold the sum of the digits raised to the power of the number of digits.

Calculate the number of digits in num and store it in a variable len.

Calculate the Armstrong Sum:

While num is not 0:

Extract the last digit of num (using num % 10) and store it in a variable digit.

Raise digit to the power of len and add the result to sum.

Remove the last digit from num (using integer division by 10).

Compare Sum with Original Number:

Check if sum is equal to originalNumber.

If they are equal, return true; otherwise, return false.

Output the Result:

If the function isArmstrong returns true, print that the number is an Armstrong number.

If the function isArmstrong returns false, print that the number is not an Armstrong number.

Implementation details:

```
import java.util.Scanner;

public class armstrong {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter an integer: ");
        int num = sc.nextInt();
        sc.close();

        if (isarmstrong(num)) {
            System.out.println(num + " is an Armstrong number.");
        } else {
            System.out.println(num + " is not an Armstrong number.");
        }
    }

    public static  boolean isarmstrong(int num) {
        int originalNumber = num;
        int sum = 0;
        int len = String.valueOf(num).length();

        while (num != 0) {
            int digit = num % 10;
            sum += Math.pow(digit, len);
            num /= 10;
        }

        return sum == originalNumber;
    }
}
```

Output:

```
Enter an integer: 153
153 is an Armstrong number.
```

```
Enter an integer: 1634
1634 is an Armstrong number.
```

```
Enter an integer: 101
101 is not an Armstrong number.
```

Conclusion:

Learned To create Methods and using predefined method using scanner class. Also learned to implement methods taking user data.

Date: _____ **Signature of faculty in-charge**

Post Lab Descriptive Questions:

Q.1 Write a program to find the perfect numbers between the range.

```
import java.util.Scanner;

public class perfect {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);

      System.out.print("Enter the lower bound of the range: ");
      int low = sc.nextInt();

      System.out.print("Enter the upper bound of the range: ");
      int up = sc.nextInt();

      sc.close();

      System.out.println("Perfect numbers between " + low + " and " + up + ":");
      for (int i = low; i <= up; i++) {
         if (isPerfect(i)) {
            System.out.println(i);
         }
      }
   }

   public static boolean isPerfect(int num) {
      if (num < 2) {
         return false;
      }

      int sum = 1;
      for (int i = 2; i <= num / 2; i++) {
         if (num % i == 0) {
            sum += i;
         }
      }
```

```
        return sum == num;
    }
}
```

Q.2 Write a program to check whether the entered year is a leap year or not.

```
import java.util.Scanner;
public class leapyear {
public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);

    System.out.print("Enter a year: ");
    int year = sc.nextInt();

    sc.close();

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

public static boolean isLeapYear(int year) {
    if (year % 4 == 0) {
        if (year % 100 == 0) {
            if (year % 400 == 0) {
                return true;
            } else {
                return false;
            }
        } else {
            return true;
        }
    } else {
        return false;
    }
```

```
  }
}
```

Q.3 Write a program to find gcd and lcm of two numbers(find gcd using recursive function).

```
import java.util.Scanner;

public class GCDAndLCM {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        System.out.print("Enter the first number: ");
        int num1 = sc.nextInt();

        System.out.print("Enter the second number: ");
        int num2 = sc.nextInt();

        sc.close();

        int gcd = findGCD(num1, num2);
        int lcm = findLCM(num1, num2, gcd);

        System.out.println("GCD of " + num1 + " and " + num2 + " is:
" + gcd);
        System.out.println("LCM of " + num1 + " and " + num2 + " is:
" + lcm);
    }

    public static int findGCD(int a, int b) {
        if (b == 0) {
            return a;
        }
        return findGCD(b, a % b);
    }

    public static int findLCM(int a, int b, int gcd) {
```

```
        return (a * b) / gcd;
    }
}
```

Output:

```
C:\Users\Saish\OneDrive\Desktop\C\java>java perfect.java
Enter the lower bound of the range: 5
Enter the upper bound of the range: 100
Perfect numbers between 5 and 100:
6
28
```

```
C:\Users\Saish\OneDrive\Desktop\C\java>java leapyear.java
Enter a year: 2024
2024 is a leap year.

C:\Users\Saish\OneDrive\Desktop\C\java>java leapyear.java
Enter a year: 2023
2023 is not a leap year.

C:\Users\Saish\OneDrive\Desktop\C\java>
```

```
C:\Users\Saish\OneDrive\Desktop\C\java>java gcdlcm.java
Enter the first number: 5
Enter the second number: 35
GCD of 5 and 35 is: 5
LCM of 5 and 35 is: 35

C:\Users\Saish\OneDrive\Desktop\C\java>java gcdlcm.java
Enter the first number: 3
Enter the second number: 69
GCD of 3 and 69 is: 3
LCM of 3 and 69 is: 69
```