**Batch: Roll No.:**

**Experiment / assignment / tutorial No.01**

**Grade: AA / AB / BB / BC / CC / CD /DD**

**Signature of the Staff In-charge with date**

|  |
| --- |
| **TITLE : Perfect Number** |

**AIM:** Define a class Perfect which accepts the range of numbers from the user. Create a static function check\_per , which checks if the number is a perfect number or not and sends the result back to the main function which counts and displays the perfect numbers within that range.

Variations :

Implementation of Program with One class

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

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Expected OUTCOME of Experiment:**

**CO2:** Explore arrays, vectors, classes and objects in C++ and Java **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Books/ Journals/ Websites referred:**

1. E. Balagurusamy , “Programming with Java” McGraw-Hill.
2. Sachin Malhotra, Saurabh Choudhary, “Programming in Java”, Oxford Publications.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**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();

**Class Diagram:**

**Algorithm:**

1. Start

2. Define a static function isPerfect(num)

a. Set sum = 0

b. For i from 1 to num/2 do steps c and d

c. If num is divisible by i, then add i to sum

d. End If

e. If sum is equal to num, return true

f. Otherwise, return false

3. Define a non-static function findPerfectNumbers(start, end)

a. Set count = 0

b. Print "Perfect numbers within the range start to end:"

c. For i from start to end do steps d and e

d. If isPerfect(i) is true, then

i. Print i

ii. Increment count by 1

e. End If

f. Print "Total perfect numbers found: count"

4. Define the main function

a. Create a Scanner object named scanner to read input

b. Print "Enter the starting range: "

c. Read start from the user using scanner

d. Print "Enter the ending range: "

e. Read end from the user using scanner

f. Create an object perfectObj of class Perfect

g. Call the findPerfectNumbers method of perfectObj with start and end as arguments

h. Print "Using static method:"

i. For i from start to end do steps j and k

j. If isPerfect(i) is true, then

i. Print i

k. End If

5. End

**Implementation details:**

import java.util.Scanner;

class Perfect

{

    static boolean isPerfect(int num)

    {

        int sum = 0;

        for (int i = 1; i <= num / 2; i++)

        {

            if (num % i == 0)

            {

                sum += i;

            }

        }

        return sum == num;

    }

    void findPerfectNumbers(int start, int end)

    {

        int count = 0;

        System.out.println("Perfect numbers within the range " + start + " to " + end + ":");

        for (int i = start; i <= end; i++) {

            if (isPerfect(i)) {

                System.out.print(i + " ");

                count++;

            }

        }

        System.out.println("\nTotal perfect numbers found: " + count);

    }

    public static void main(String[] args)

    {

        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter the starting range: ");

        int start = scanner.nextInt();

        System.out.print("Enter the ending range: ");

        int end = scanner.nextInt();

        // Using non-static method to find and display perfect numbers

        Perfect perfectObj = new Perfect();

        perfectObj.findPerfectNumbers(start, end);

        // Using static method to find and display perfect numbers

        System.out.println("Using static method:");

        for (int i = start; i <= end; i++)

        {

            if (Perfect.isPerfect(i))

            {

                System.out.print(i + " ");

            }

        }

    }

}

**Output:**

A screenshot of a computer

Description automatically generated

**Conclusion:**

Hence, we learned how to find perfect number program using different variations like static and non-static inside and outside class.

**Date: 05/08/2023 Signature of faculty in-charge**

**Post Lab Descriptive Questions:**

Q.1 Write a program to find the area and circumference of a circle using two classes.

**Ans:**

import java.util.Scanner;

class Circle {

private double radius;

// Constructor to initialize the radius

public Circle(double radius) {

this.radius = radius;

}

// Method to calculate the area of the circle

public double calculateArea() {

return Math.PI \* radius \* radius;

}

// Method to calculate the circumference of the circle

public double calculateCircumference() {

return 2 \* Math.PI \* radius;

}

}

public class Main {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter the radius of the circle: ");

double radius = scanner.nextDouble();

// Create an object of the Circle class

Circle circle = new Circle(radius);

// Calculate the area and circumference using the methods of the Circle class

double area = circle.calculateArea();

double circumference = circle.calculateCircumference();

System.out.println("Area of the circle: " + area);

System.out.println("Circumference of the circle: " + circumference);

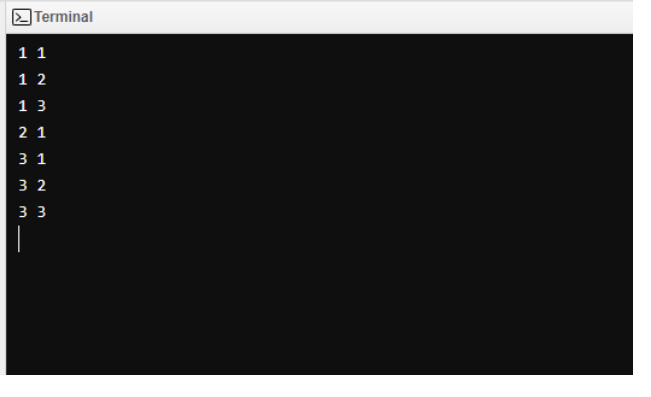
}

}

Q.2 Write the output of following program

1. **public** **class** BreakExample2 {
2. **public** **static** **void** main(String[] args) {
3. //outer loop
4. **for**(**int** i=1;i<=3;i++){
5. //inner loop
6. **for**(**int** j=1;j<=3;j++){
7. **if**(i==2&&j==2){
8. //using break statement inside the inner loop
9. **break**;
10. }
11. System.out.println(i+" "+j);
12. }
13. }
14. }
15. }

**Output:**



Q.3 Why is Java known as a platform independent language?

**Ans:**

Java's platform independence is achieved by compiling the Java code into bytecode, which is an intermediate representation that is not tied to any specific platform. The bytecode is then executed by the Java Virtual Machine (JVM), which is available on various platforms. The JVM interprets the bytecode and translates it into native machine code at runtime, enabling Java programs to run on any platform with a compatible JVM.

One of the key factors contributing to Java's platform independence is the absence of platform-specific features in its core libraries and language constructs. Instead, Java abstracts any platform-specific functionality and provides it through the Java API, ensuring consistent behavior across different platforms. This approach allows Java programs to be written once and run anywhere, making Java a truly cross-platform language.

Q.4 Write a recursive static method for calculation of gcd of a number.

**Ans:**

public class GCDCalculator

{

public static int calculateGCD(int num1, int num2)

{

if (num2 == 0)

{

return num1;

}

else

{

return calculateGCD(num2, num1 % num2);

}

}

public static void main(String[] args) {

int num1 = 48;

int num2 = 18;

int gcd = calculateGCD(num1, num2);

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

}

}