**Batch: C3 Roll No.: 16010123217**

**Experiment / assignment / tutorial No. 2**

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

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

|  |
| --- |
| **TITLE : Control Statements** |

**AIM:**

Write a Java program to generate and show all Kaprekar numbers less than 1000.

In number theory, a Kaprekar number for a given base is a non-negative integer, the representation of whose square in that base can be split into two parts that add up to the original number again. For instance, 45 is a Kaprekar number, because 45^2 = 2025 and 20 + 25 = 45.

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

**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:**

Java basic constructs (like if else statement, control structures, and data types

Programming languages provide various control structures that allow for more complicated execution paths.

A loop statement allows us to execute a statement or group of statements multiple times and following is the general form of a loop statement in most of the programming languages −

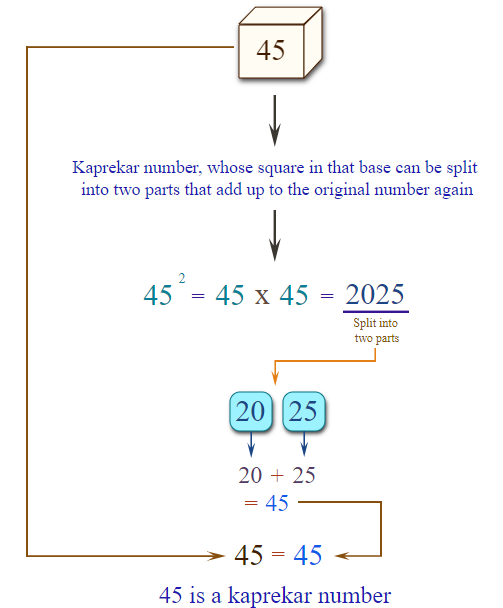
|  |  |
| --- | --- |
| **Sr.No.** | **Loop & Description** |
| 1 | [**while loop**](https://www.tutorialspoint.com/java/java_while_loop.htm)  Repeats a statement or group of statements while a given condition is true. It tests the condition before executing the loop body. |
| 2 | [**for loop**](https://www.tutorialspoint.com/java/java_for_loop.htm)  Execute a sequence of statements multiple times and abbreviates the code that manages the loop variable. |
| 3 | [**do...while loop**](https://www.tutorialspoint.com/java/java_do_while_loop.htm)  Like a while statement, except that it tests the condition at the end of the loop body. |

**Loop Control Statements**

Loop control statements change execution from its normal sequence. When execution leaves a scope, all automatic objects that were created in that scope are destroyed.

Java supports the following control statements. Click the following links to check their details.

|  |  |
| --- | --- |
| **Sr.No.** | **Control Statement & Description** |
| 1 | [**break statement**](https://www.tutorialspoint.com/java/java_break_statement.htm)  Terminates the loop or switch statement and transfers execution to the statement immediately following the loop or switch. |
| 2 | [**continue statement**](https://www.tutorialspoint.com/java/java_continue_statement.htm)  Causes the loop to skip the remainder of its body and immediately retest its condition prior to reiterating. |



In number theory, a Kaprekar number for a given base is a non-negative integer, the representation of whose square in that base can be split into two parts that add up to the original number again. For instance, 45 is a Kaprekar number, because 452 = 2025 and 20 + 25 = 45.

**Algorithm:**

1. Define class Kaprekar

2. Define static method kaprekar(int n):

1. Initialize count = 0, num = 0, s = 0, and temp = n

2. Calculate the number of digits in n:

While n != 0:

n = n / 10

Increment count

3. Calculate square of n as n = temp \* temp

4. Extract the right part of the square:

For i = 0 to count - 1:

num = n % (10^count)

5. Extract the left part of the square:

s = n / (10^count)

6. Return (s + num)

3. Define main method:

1. Print "The Kaprekar Numbers less than 1000 are:"

2. Iterate from i = 1 to 999:

If kaprekar(i) == i:

Print i

4. End program

**Implementation details:**

import java.util.*\**;

public class Kaprekar

{

    static int kaprekar(int n)

    {

        int count = 0, num = 0, temp=n, s = 0;

        while(n!=0)

        {

            n=n/10;

            count++;

        }

        n = temp\*temp;

        for(int i=0;i<count;i++)

        {

            num = (int)(n % (Math.pow(10,count)));

        }

        s = (int) (n/(Math.pow(10,count)));

        return (s+num);

    }

    public static void main(String [] args)

    {

        System.out.println("The Kaprekar Numbers less than 1000 are:");

        for(int i = 1; i<1000; i++){

        if(kaprekar(i) == i)

        {

            System.out.println(i);

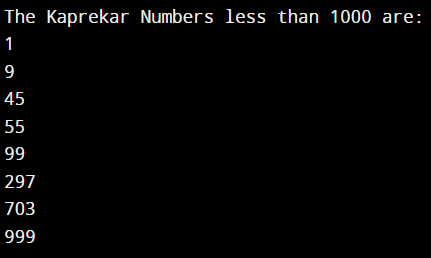
        }

        }

    }

}

**Output:**



**Conclusion:**

**This program not only helps in understanding the concept of Kaprekar numbers but also reinforces basic programming principles such as loops, conditionals, and arithmetic operations.**

**Date: 14/08/24 Signature of faculty in-charge**

**Post Lab Descriptive Questions:**

Q.1 Write a program to find the largest of three numbers using the if-else construct.

Ans.

import java.util.Scanner;

public class Numbers {

    double num1, num2, num3;

    public Numbers(double a, double b, double c) {

        num1 = a;

        num2 = b;

        num3 = c;

    }

    public double largest() {

        if (num1 >= num2 && num1 >= num3) {

            return num1;

        } else if (num2 >= num1 && num2 >= num3) {

            return num2;

        } else {

            return num3;

        }

    }

    public static void main(String[] args) {

        Scanner s1 = new Scanner(System.in);

        System.out.print("Enter the first number: ");

        double num1 = s1.nextDouble();

        System.out.print("Enter the second number: ");

        double num2 = s1.nextDouble();

        System.out.print("Enter the third number: ");

        double num3 = s1.nextDouble();

        Numbers large = new Numbers(num1, num2, num3);

        System.out.println("The largest number is: " + large.largest());

    }

}

Q.2 Write a program to determine the sum of the following series for a given value of n:1+½+⅓+....+1/n

Ans.

import java.util.Scanner;

public class SeriesSum {

    public static void main(String[] args) {

        Scanner s1 = new Scanner(System.in);

        System.out.print("Enter the value of n: ");

        int n = s1.nextInt();

        double sum = 0.0;

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

            sum += 1.0 / i;

        }

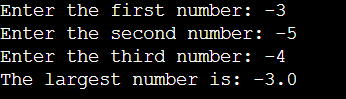
        System.out.println("The sum of the series is: " + sum);

    }

}

**Output:**

**1)**



**2)**

