Name – Piyush Kandhwey

Reg no – RA2411028010105

# Level 1 Practice Programs

1. Write a program to input the Principal, Rate, and Time values and calculate Simple Interest.

**Hint =>**

1. Simple Interest = Principal \* Rate \* Time / 100
2. Take user input for principal, rate, time
3. Write a method to calculate the simple interest given principle, rate and time as parameters
4. Output “The Simple Interest is \_\_\_ for Principal \_\_\_, Rate of Interest \_\_\_ and Time \_\_\_”

Solution –

Program –

import java.util.Scanner;

public class SimpleInterestCalculator {

public static double calculateSimpleInterest(double principal, double rate, double time) {

return (principal \* rate \* time) / 100;

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

double principal = sc.nextDouble();

double rate = sc.nextDouble();

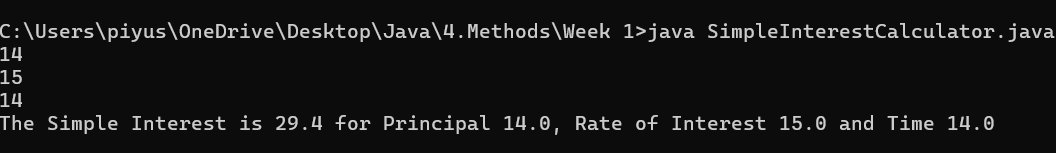
double time = sc.nextDouble();

double interest = calculateSimpleInterest(principal, rate, time);

System.out.println("The Simple Interest is " + interest + " for Principal " + principal + ", Rate of Interest " + rate + " and Time " + time);

}

}



1. Create a program to find the maximum number of handshakes among N number of students.

**Hint =>**

1. Get integer input for number of students
2. Use the combination = (n \* (n - 1)) / 2 formula to calculate the maximum number of possible handshakes.
3. Write a method to use the combination formulae to calculate the number of handshakes

Solution –

Program –

import java.util.Scanner;

public class HandshakeCalculator {

public static int calculateHandshakes(int n) {

return (n \* (n - 1)) / 2;

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

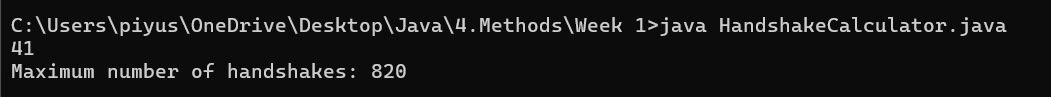
int students = sc.nextInt();

int handshakes = calculateHandshakes(students);

System.out.println("Maximum number of handshakes: " + handshakes);

}

}



1. Create a program to find the maximum number of handshakes among N number of students.

**Hint =>**

1. Get integer input for numberOfStudents variable.
2. Use the combination = (n \* (n - 1)) / 2 formula to calculate the maximum number of possible handshakes.
3. Display the number of possible handshakes.

Solution –

Program –

import java.util.Scanner;

public class HandshakeCalculator {

public static int calculateHandshakes(int n) {

return (n \* (n - 1)) / 2;

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

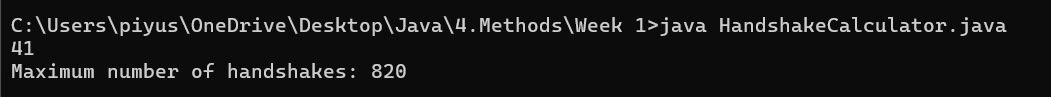
int students = sc.nextInt();

int handshakes = calculateHandshakes(students);

System.out.println("Maximum number of handshakes: " + handshakes);

}

}



1. An athlete runs in a triangular park with sides provided as input by the user in meters. If the athlete wants to complete a 5 km run, then how many rounds must the athlete complete

**Hint =>**

1. Take user input for 3 sides of a triangle
2. The perimeter of a triangle is the addition of all sides and rounds is distance/perimeter
3. Write a Method to compute the number of rounds user needs to do to complete 5km run

Solution –

Program –

import java.util.Scanner;

public class TriangularParkRun {

public static double calculateRounds(double a, double b, double c) {

double perimeter = a + b + c;

return 5000 / perimeter;

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

double side1 = sc.nextDouble();

double side2 = sc.nextDouble();

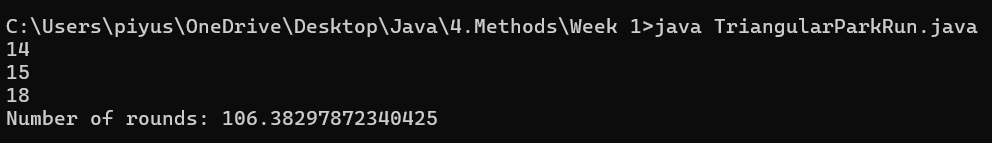
double side3 = sc.nextDouble();

double rounds = calculateRounds(side1, side2, side3);

System.out.println("Number of rounds: " + rounds);

}

}



1. Write a program to check whether a number is positive, negative, or zero.

**Hint =>** Get integer input from the user. Write a Method to return -1 for negative number, 1 for positive number and 0 if number is zero

Solution –

Program –

import java.util.Scanner;

public class NumberSignChecker {

public static int checkNumber(int num) {

if (num > 0) return 1;

else if (num < 0) return -1;

else return 0;

}

public static void main(String[] args) {

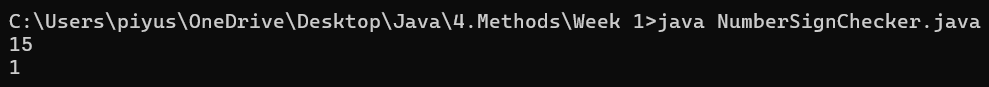
Scanner sc = new Scanner(System.in);

int number = sc.nextInt();

System.out.println(checkNumber(number));

}

}



1. Write a program SpringSeason that takes two int values month and day from the command line and prints “Its a Spring Season” otherwise prints “Not a Spring Season”.

**Hint =>** Spring Season is from March 20 to June 20. Write a Method to check for Spring season and return a boolean true or false

Solution –

Program –

import java.util.Scanner;

public class SpringSeason {

public static boolean isSpring(int month, int day) {

return (month == 3 && day >= 20) || (month == 6 && day <= 20) || (month > 3 && month < 6);

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

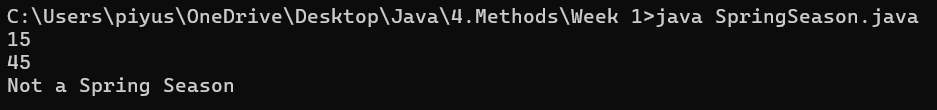
int month = sc.nextInt();

int day = sc.nextInt();

System.out.println(isSpring(month, day) ? "Its a Spring Season" : "Not a Spring Season");

}

}

****

1. Write a program to find the sum of n natural numbers using loop

**Hint =>** Get integer input from the user. Write a Method to find the sum of n natural numbers using loop

Solution –

Program –

import java.util.Scanner;

public class NaturalNumberSum {

public static int findSum(int n) {

int sum = 0;

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

sum += i;

}

return sum;

}

public static void main(String[] args) {

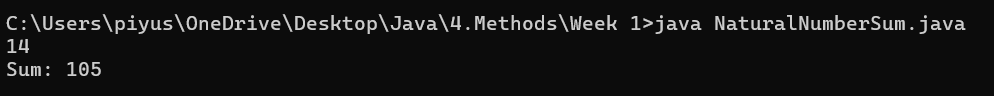
Scanner sc = new Scanner(System.in);

int n = sc.nextInt();

System.out.println("Sum: " + findSum(n));

}

}



1. Write a program to find the smallest and the largest of the 3 numbers.

**Hint =>**

1. Take user input for 3 numbers
2. Write a single method to find the smallest and largest of the three numbers

***public static int[] findSmallestAndLargest(int number1, int number2, int number3)***

Solution –

Program –

import java.util.Scanner;

public class MinMaxFinder {

public static int[] findSmallestAndLargest(int a, int b, int c) {

int smallest = Math.min(a, Math.min(b, c));

int largest = Math.max(a, Math.max(b, c));

return new int[]{smallest, largest};

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int a = sc.nextInt();

int b = sc.nextInt();

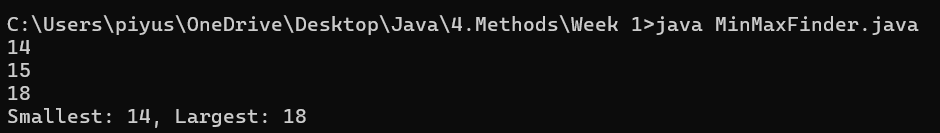
int c = sc.nextInt();

int[] result = findSmallestAndLargest(a, b, c);

System.out.println("Smallest: " + result[0] + ", Largest: " + result[1]);

}

}



1. Write a program to take 2 numbers and print their quotient and reminder

**Hint =>**

1. Take user input as integer
2. Use division operator (/) for quotient and moduli operator (%) for reminder
3. Write Method to find the reminder and the quotient of a number

***public static int[] findRemainderAndQuotient(int number, int divisor)***

Solution –

Program –

import java.util.Scanner;

public class DivisionCalculator {

public static int[] findRemainderAndQuotient(int number, int divisor) {

return new int[]{number % divisor, number / divisor};

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int number = sc.nextInt();

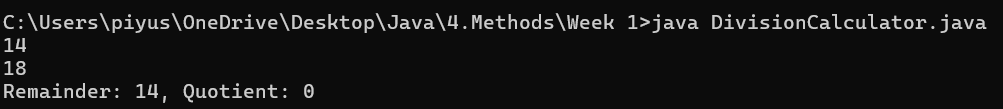
int divisor = sc.nextInt();

int[] result = findRemainderAndQuotient(number, divisor);

System.out.println("Remainder: " + result[0] + ", Quotient: " + result[1]);

}

}



1. Create a program to divide N number of chocolates among M children. Print the number of chocolates each child will get and also the remaining chocolates

**Hint =>**

1. Get an integer value from user for the numberOfchocolates and numberOfChildren.
2. Write the method to find the number of chocolates each child gets and number of remaining chocolates

***public static int[] findRemainderAndQuotient(int number, int divisor)***

Solution –

Program –

import java.util.Scanner;

public class ChocolateDivider {

public static int[] findRemainderAndQuotient(int number, int divisor) {

return new int[]{number % divisor, number / divisor};

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int chocolates = sc.nextInt();

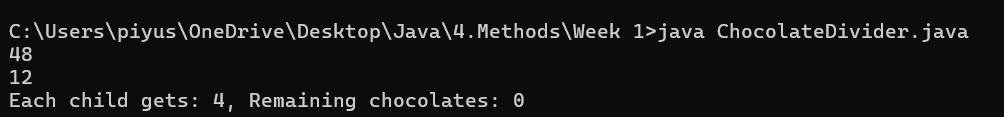
int children = sc.nextInt();

int[] result = findRemainderAndQuotient(chocolates, children);

System.out.println("Each child gets: " + result[1] + ", Remaining chocolates: " + result[0]);

}

}

****

1. Write a program calculate the wind chill temperature given the temperature and wind speed

**Hint =>**

1. Write a method to calculate the wind chill temperature using the formula

***public double calculateWindChill(double temperature, double windSpeed)***

Solution –

Program –

import java.util.Scanner;

public class WindChillCalculator {

public static double calculateWindChill(double temperature, double windSpeed) {

return 35.74 + 0.6215 \* temperature + (0.4275 \* temperature - 35.75) \* Math.pow(windSpeed, 0.16);

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

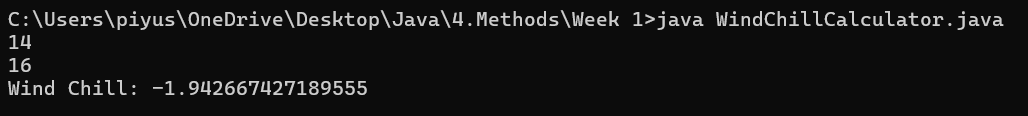
double temperature = sc.nextDouble();

double windSpeed = sc.nextDouble();

System.out.println("Wind Chill: " + calculateWindChill(temperature, windSpeed));

}

}



1. Write a program to calculate various trigonometric functions using Math class given an angle in degrees

**Hint =>**

1. Method to calculate various trigonometric functions, Firstly convert to radians and then use Math function to find sine, cosine and tangent.

***public double[] calculateTrigonometricFunctions(double angle)***

Solution –

Program –

import java.util.Scanner;

public class TrigFunctions {

public static double[] calculateTrigonometricFunctions(double angle) {

double radians = Math.toRadians(angle);

return new double[]{Math.sin(radians), Math.cos(radians), Math.tan(radians)};

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

double angle = sc.nextDouble();

double[] trigValues = calculateTrigonometricFunctions(angle);

System.out.println("Sin: " + trigValues[0]);

System.out.println("Cos: " + trigValues[1]);

System.out.println("Tan: " + trigValues[2]);

}

}

