**5.7 Values of** x**:** a) x = Math.abs(7.5); Result: x = 7.5 (Absolute value of 7.5 is itself). b) x = Math.floor(7.5); Result: x = 7.0 (Floor gives the largest integer less than or equal to 7.5). c) x = Math.abs(0.0); Result: x = 0.0 (Absolute value of 0 is 0). d) x = Math.ceil(0.0); Result: x = 0.0 (Ceiling gives the smallest integer greater than or equal to 0). e) x = Math.abs(-6.4); Result: x = 6.4 (Absolute value of -6.4 is 6.4). f) x = Math.ceil(-6.4); Result: x = -6.0 (Ceiling gives the smallest integer greater than or equal to -6.4). g) x = Math.ceil(-Math.abs(-8 + Math.floor(-5.5))); Intermediate steps:

* Math.floor(-5.5) = -6.0
* -8 + (-6.0) = -14.0
* Math.abs(-14.0) = 14.0
* Math.ceil(-14.0) = -14.0 Result: x = -14.0.

5.8) import java.util.Scanner;

public class ParkingCharges {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

double totalCharges = 0.0;

while (true) {

System.out.print("Enter hours parked (or -1 to end): ");

int hours = input.nextInt();

if (hours == -1) break;

double charge = calculateCharges(hours);

System.out.printf("Charge for customer: $%.2f%n", charge);

totalCharges += charge;

}

System.out.printf("Total receipts: $%.2f%n", totalCharges);

}

public static double calculateCharges(int hours) {

double charge = 2.0;

if (hours > 3) {

charge += 0.5 \* (hours - 3);

}

return Math.min(charge, 10.0);

}

}

5.9) import java.util.Scanner;

public class RoundingNumbers {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

while (true) {

System.out.print("Enter a number to round (or -1 to end): ");

double x = input.nextDouble();

if (x == -1) break;

long roundedValue = Math.round(x); // Using Math.round for simplicity

System.out.printf("Original: %.2f, Rounded: %d%n", x, roundedValue);

}

}

}

5.10 ) public class RoundingDecimals {

public static void main(String[] args) {

double[] numbers = {5.6789, 3.14159, 9.876}; // Example values

for (double number : numbers) {

System.out.printf("Original: %.4f%n", number);

System.out.printf("Rounded to integer: %.0f%n", roundToInteger(number));

System.out.printf("Rounded to tenths: %.1f%n", roundToTenths(number));

System.out.printf("Rounded to hundredths: %.2f%n", roundToHundredths(number));

System.out.printf("Rounded to thousandths: %.3f%n%n", roundToThousandths(number));

}

}

public static double roundToInteger(double number) {

return Math.floor(number + 0.5);

}

public static double roundToTenths(double number) {

return Math.floor(number \* 10 + 0.5) / 10;

}

public static double roundToHundredths(double number) {

return Math.floor(number \* 100 + 0.5) / 100;

}

public static double roundToThousandths(double number) {

return Math.floor(number \* 1000 + 0.5) / 1000;

}

}

5.12) // a) 1 ≤ n ≤ 2

int n = 1 + (int)(Math.random() \* 2);

// b) 1 ≤ n ≤ 100

n = 1 + (int)(Math.random() \* 100);

// c) 0 ≤ n ≤ 9

n = (int)(Math.random() \* 10);

// d) 1000 ≤ n ≤ 1112

n = 1000 + (int)(Math.random() \* 113);

// e) –1 ≤ n ≤ 1

n = -1 + (int)(Math.random() \* 3);

// f) –3 ≤ n ≤ 11

n = -3 + (int)(Math.random() \* 15);

5.13) // a) 2, 4, 6, 8, 10

int n = 2 \* (1 + (int)(Math.random() \* 5));

// b) 3, 5, 7, 9, 11

n = 3 + 2 \* (int)(Math.random() \* 5);

// c) 6, 10, 14, 18, 22

n = 6 + 4 \* (int)(Math.random() \* 5);