Patuakhali Science and Technology University

Assignment on

"Liang Book chapter 3 Exercise Solve"

Course Code: CCE-122
Course Title: Object Oriented Programming

Level - I; Semester - II

Submitted By

Name: M.D. Sakibul Islam Shovon
ID: 2302056
REG: 11834
Session: 2023-2024
Faculty of Computer Science and Engineering

Submitted To

Prof. Dr. Md. Samsuzzaman
Professor of Computer and Communication Engineering Department
Faculty of Computer Science and Engineering

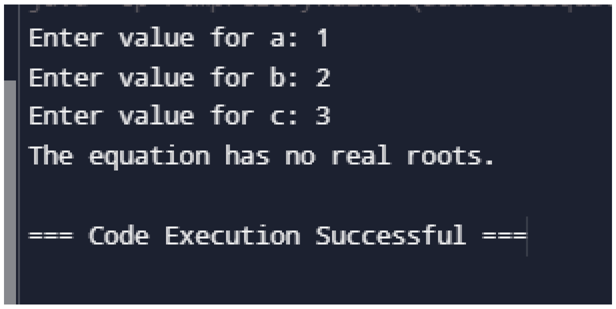
Liang Book chapter 3 Exercise Solve:

3.1 (Algebra: solve quadratic equations) The two roots of a quadratic equation
ax2 + bx + c = 0 can be obtained using the following formula:

$r_1 = \frac{-b+\sqrt{b^2-4\text{ac}}}{2a}$ and $r_2 = \frac{-b-\sqrt{b^2-4\text{ac}}}{2a}$

b2 - 4ac is called the discriminant of the quadratic equation. If it is positive, the equation has two real roots. If it is zero, the equation has one root. If it is negative, the equation has no real roots. Write a program that prompts the user to enter values for a, b, and c and displays the result based on the discriminant. If the discriminant is positive, display two roots. If the discriminant is 0, display one root. Otherwise, display “The equation has no real roots.”

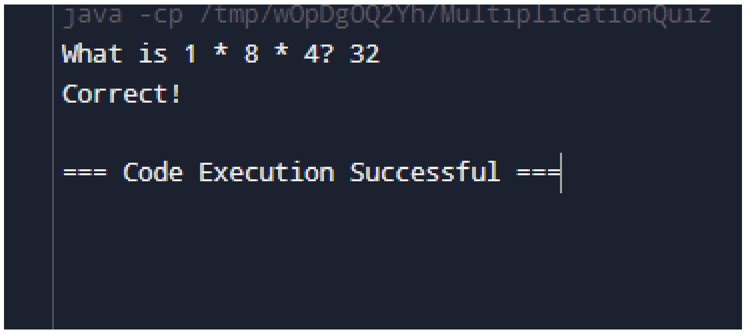
```
import java.util.Scanner;


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


    System.out.print("Enter value for a: ");
    double a = scanner.nextDouble();


    System.out.print("Enter value for b: ");
    double b = scanner.nextDouble();
    System.out.print("Enter value for c: ");
    double c = scanner.nextDouble();


    double discriminant = b * b - 4 * a * c;


    if (discriminant > 0) {
      double root1 = (-b + Math.pow(discriminant, 0.5)) / (2 * a);
      double root2 = (-b - Math.pow(discriminant, 0.5)) / (2 * a);
      System.out.println("The equation has two real roots: " + root1 + " and
" + root2);
    } else if (discriminant == 0) {
      double root = -b / (2 * a);
      System.out.println("The equation has one real root: " + root);
    } else {
      System.out.println("The equation has no real roots.");
    }
  }
}
```



```
Enter value for a: 1
Enter value for b: 2
Enter value for c: 3
The equation has no real roots.

=== Code Execution Successful ===
```

3.2 (Game: multiply three numbers) The program in Listing 3.1, AdditionQuiz.java, generates two integers and prompts the user to enter the product of these two integers. Revise the program to generate three single-digit integers and prompt the user to enter the multiplication of these three integers.

```
import java.util.Scanner;


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


    int num1 = (int) (Math.random() * 10);
    int num2 = (int) (Math.random() * 10);
    int num3 = (int) (Math.random() * 10);


    System.out.print("What is " + num1 + " * " + num2 + " * " + num3 + "? 
");
    int answer = scanner.nextInt();


    int correctAnswer = num1 * num2 * num3;
    if (answer == correctAnswer) {
      System.out.println("Correct!");
    } else {
      System.out.println("Sorry, the correct answer is " + correctAnswer);
    }
  }
}
```



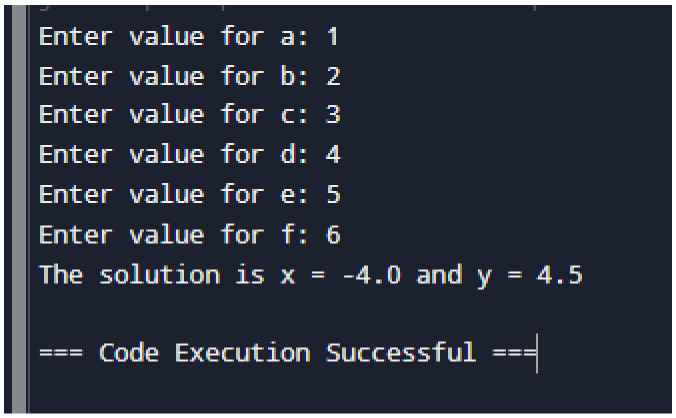
```
java -cp /tmp/wOpDgOQ2Yh/MultiplicationQuiz
What is 1 * 8 * 4? 32
Correct!

=== Code Execution Successful ===
```

3.3 (Algebra: solve 2 * 2 linear equations) A linear equation can be solved using Cramer’s rule given in Programming Exercise 1.13. Write a program that prompts the user to enter a, b, c, d, e, and f and displays the result. If ad - bc is 0, report that “The equation has no solution.”

```
import java.util.Scanner;


public class LinearEquationSolver {
  public static void main(String[] args) {
    Scanner scanner = new Scanner(System.in);
```

```
System.out.print("Enter value for a: ");
double a = scanner.nextDouble();

System.out.print("Enter value for b: ");
double b = scanner.nextDouble();

System.out.print("Enter value for c: ");
double c = scanner.nextDouble();

System.out.print("Enter value for d: ");
double d = scanner.nextDouble();

System.out.print("Enter value for e: ");
double e = scanner.nextDouble();

System.out.print("Enter value for f: ");
double f = scanner.nextDouble();

double denominator = a * d - b * c;

if (denominator == 0) {
  System.out.println("The equation has no solution.");
} else {
  double x = (e * d - b * f) / denominator;
```

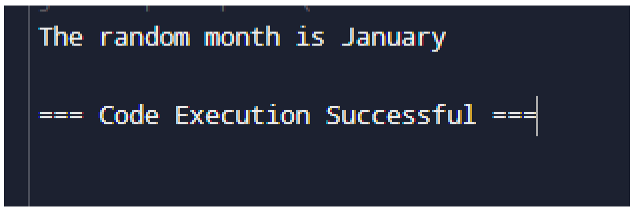
```
    double y = (a * f - e * c) / denominator;


    System.out.println("The solution is x = " + x + " and y = " + y);
   }
  }
}
```



```
Enter value for a: 1
Enter value for b: 2
Enter value for c: 3
Enter value for d: 4
Enter value for e: 5
Enter value for f: 6
The solution is x = -4.0 and y = 4.5

=== Code Execution Successful ===
```

3.4 (Random month) Write a program that randomly generates an integer between 1 and 12 and displays the English month names January, February, . . . , December for the numbers 1, 2, . . . , 12, accordingly.

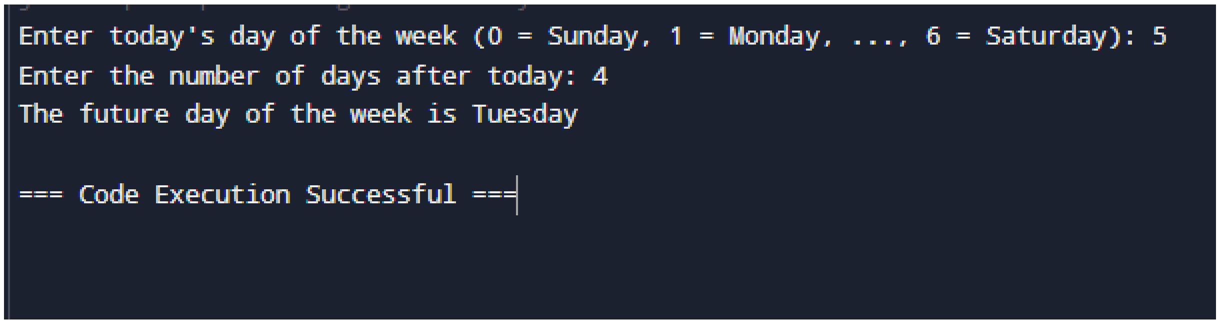
```
import java.util.Random;


public class RandomMonth {
  public static void main(String[] args) {
  String[] monthNames = {"January", "February", "March", "April", "May",
"June", "July", "August", "September", "October", "November",
"December"};

  Random random = new Random();
  int monthNumber = random.nextInt(12) + 1;

  String monthName = monthNames[monthNumber - 1];

  System.out.println("The random month is " + monthName);
  }
}
```



```
The random month is January

=== Code Execution Successful ===
```

3.5 (Find future dates) Write a program that prompts the user to enter an integer for today's day of the week (Sunday is 0, Monday is 1, . . . , and Saturday is 6). Also prompt the user to enter the number of days after today for a future day and display the future day of the week. Here is a sample run:

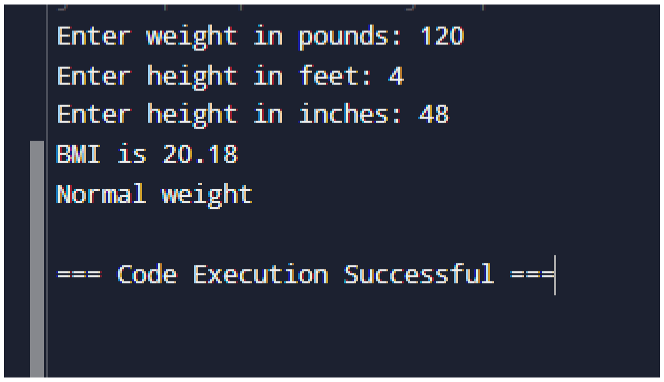
```
 import java.util.Scanner;

 public class FutureDay {
  public static void main(String[] args) {
    String[] dayNames = {"Sunday", "Monday", "Tuesday", "Wednesday",
"Thursday", "Friday", "Saturday"};

    Scanner scanner = new Scanner(System.in);

    System.out.print("Enter today's day of the week (0 = Sunday, 1 =
Monday, ..., 6 = Saturday): ");
    int today = scanner.nextInt();

    System.out.print("Enter the number of days after today: ");
    int daysAfterToday = scanner.nextInt();

    int futureDay = (today + daysAfterToday) % 7;

    System.out.println("The future day of the week is " +
dayNames[futureDay]);
  }
}
```



```
Enter today's day of the week (0 = Sunday, 1 = Monday, ..., 6 = Saturday): 5
Enter the number of days after today: 4
The future day of the week is Tuesday

=== Code Execution Successful ===
```

3.6 (Health application: BMI) Revise Listing 3.4, ComputeAndInterpretBMI.java, to let the user enter weight, feet, and inches. For example, if a person is 5 feet and 10 inches, you will enter 5 for feet and 10 for inches. Here is a sample run:

```
import java.util.Scanner;


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


    System.out.print("Enter weight in pounds: ");
    double weight = scanner.nextDouble();


    System.out.print("Enter height in feet: ");
    int feet = scanner.nextInt();


    System.out.print("Enter height in inches: ");
    int inches = scanner.nextInt();


    double heightInInches = feet * 12 + inches;
    double heightInMeters = heightInInches * 0.0254;
    double bmi = weight / Math.pow(heightInMeters, 2);


    System.out.printf("BMI is %.2f\n", bmi);
    if (bmi < 18.5) {
      System.out.println("Underweight");
    } else if (bmi < 25) {
      System.out.println("Normal weight");
    } else if (bmi < 30) {
      System.out.println("Overweight");
    } else {
      System.out.println("Obese");
    }
  }
}
```



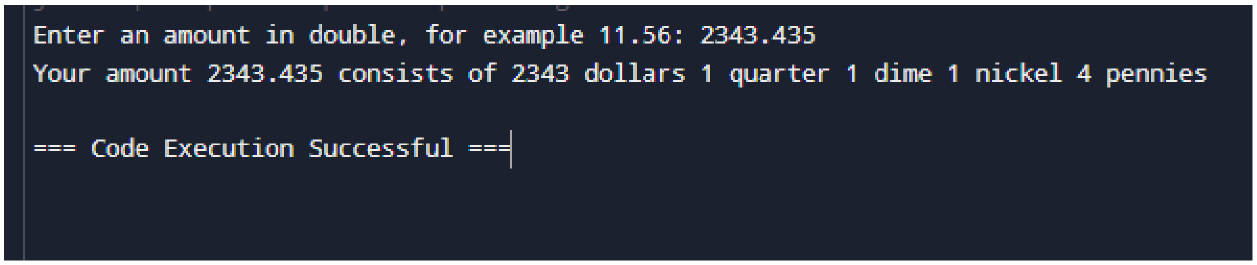
```
Enter weight in pounds: 120
Enter height in feet: 4
Enter height in inches: 48
BMI is 20.18
Normal weight

=== Code Execution Successful ===
```

3.7 (Financial application: monetary units) Modify Listing 2.10, ComputeChange. java, to display the nonzero denominations only, using singular words for single units such as 1 dollar and 1 penny, and plural words for more than one unit such as 2 dollars and 3 pennies.

```
import java.util.Scanner;

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

  System.out.print("Enter an amount in double, for example 11.56: ");
  double amount = scanner.nextDouble();

  int remainingAmount = (int) Math.round(amount * 100);

  int dollars = remainingAmount / 100;
  remainingAmount %= 100;

  int quarters = remainingAmount / 25;
  remainingAmount %= 25;

  int dimes = remainingAmount / 10;
  remainingAmount %= 10;

  int nickels = remainingAmount / 5;
  remainingAmount %= 5;

  int pennies = remainingAmount;
```

```
    System.out.print("Your amount " + amount + " consists of ");

    if (dollars > 0) {
      System.out.print(dollars + " " + (dollars == 1 ? "dollar" : "dollars") + "
");
    }

    if (quarters > 0) {
      System.out.print(quarters + " " + (quarters == 1 ? "quarter" :
"quarters") + " ");
    }

    if (dimes > 0) {
      System.out.print(dimes + " " + (dimes == 1 ? "dime" : "dimes") + " ");
    }

    if (nickels > 0) {
      System.out.print(nickels + " " + (nickels == 1 ? "nickel" : "nickels") + "
");
    }

    if (pennies > 0) {
      System.out.print(pennies + " " + (pennies == 1 ? "penny" :
"pennies"));
```

```
    }

    System.out.println();
  }
}
```

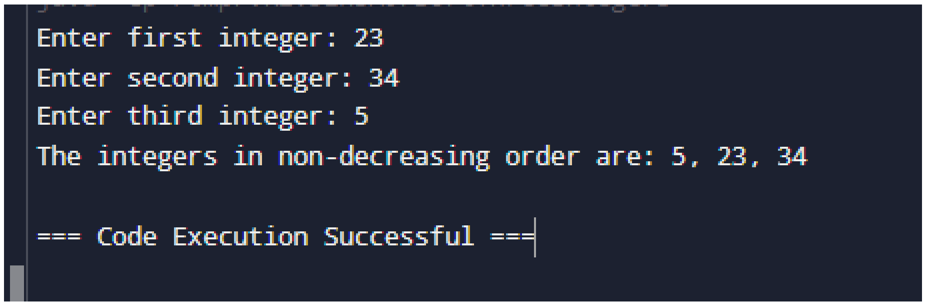


```
Enter an amount in double, for example 11.56: 2343.435
Your amount 2343.435 consists of 2343 dollars 1 quarter 1 dime 1 nickel 4 pennies

=== Code Execution Successful ===
```

3.8 (Sort three integers) Write a program that prompts the user to enter three integers and display the integers in non-decreasing order.

```
import java.util.Scanner;

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

    System.out.print("Enter first integer: ");
    int num1 = scanner.nextInt();

    System.out.print("Enter second integer: ");
    int num2 = scanner.nextInt();
    System.out.print("Enter third integer: ");
    int num3 = scanner.nextInt();

    // Sort the integers in non-decreasing order
    int temp;
    if (num1 > num2) {
      temp = num1;
      num1 = num2;
      num2 = temp;
    }
    if (num2 > num3) {
      temp = num2;
      num2 = num3;
      num3 = temp;
    }
    if (num1 > num2) {
      temp = num1;
      num1 = num2;
      num2 = temp;
    }

    System.out.println("The integers in non-decreasing order are: " +
num1 + ", " + num2 + ", " + num3);
```

```
  }
 }
```



```
Enter first integer: 23
Enter second integer: 34
Enter third integer: 5
The integers in non-decreasing order are: 5, 23, 34

=== Code Execution Successful ===
```

3.9 (Business: check ISBN-10) An ISBN-10 (International Standard Book Number) consists of 10 digits: d1d2d3d4d5d6d7d8d9d10. The last digit, d10, is a checksum, which is calculated from the other 9 digits using the following formula:

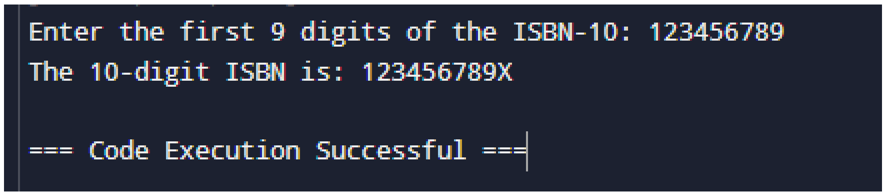
$(d_1 * 1 + d_2 * 2 + d_3 * 3 + d_4 * 4 + d_5 * 5 +$
$d_6 * 6 + d_7 * 7 + d_8 * 8 + d_9 * 9) \% 11$

If the checksum is 10, the last digit is denoted as X according to the ISBN-10 convention. Write a program that prompts the user to enter the first 9 digits and displays the 10-digit ISBN (including leading zeros). Your program should read the input as an integer.

```
import java.util.Scanner;


public class CheckISBN10 {
  public static void main(String[] args) {
    Scanner scanner = new Scanner(System.in);


    System.out.print("Enter the first 9 digits of the ISBN-10: ");
```

```java
int isbn9 = scanner.nextInt();

// Extract the individual digits from the input
int d1 = isbn9 / 100000000;
int d2 = (isbn9 / 10000000) % 10;
int d3 = (isbn9 / 1000000) % 10;
int d4 = (isbn9 / 100000) % 10;
int d5 = (isbn9 / 10000) % 10;
int d6 = (isbn9 / 1000) % 10;
int d7 = (isbn9 / 100) % 10;
int d8 = (isbn9 / 10) % 10;
int d9 = isbn9 % 10;

// Calculate the checksum
int checksum = (d1 * 1 + d2 * 2 + d3 * 3 + d4 * 4 + d5 * 5 +
        d6 * 6 + d7 * 7 + d8 * 8 + d9 * 9) % 11;

// Determine the last digit (d10)
char d10;
if (checksum == 10) {
  d10 = 'X';
} else {
  d10 = (char) (checksum + '0');
}
```

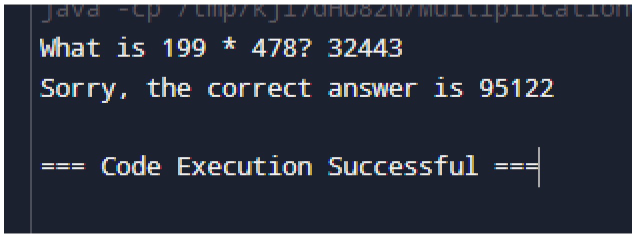
```
    // Display the 10-digit ISBN (including leading zeros)
    System.out.println("The 10-digit ISBN is: " +
                       String.format("%09d", isbn9) + d10);
  }
}
```



```
Enter the first 9 digits of the ISBN-10: 123456789
The 10-digit ISBN is: 123456789X

=== Code Execution Successful ===
```

3.10 (Game: multiplication quiz) Listing 3.3, SubtractionQuiz.java, randomly generates a subtraction question. Revise the program to randomly generate a multiplication question with two integers less than 1000.

```
import java.util.Random;
import java.util.Scanner;

public class MultiplicationQuiz {
  public static void main(String[] args) {
    Random random = new Random();
    Scanner scanner = new Scanner(System.in);

    // Generate two random integers less than 1000
    int num1 = random.nextInt(1000);
    int num2 = random.nextInt(1000);

    // Calculate the product
    int product = num1 * num2;

    // Prompt the user to answer the question
    System.out.print("What is " + num1 + " * " + num2 + "? ");
    int answer = scanner.nextInt();

    // Check if the user's answer is correct
    if (answer == product) {
      System.out.println("Correct!");
    } else {
      System.out.println("Sorry, the correct answer is " + product);
    }
  }
}
```



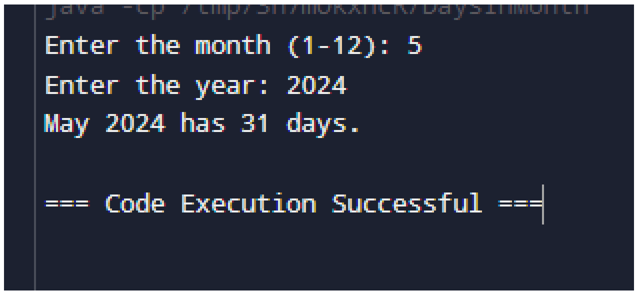
```
java -cp /tmp/kj17dHO82N/Multiplication
What is 199 * 478? 32443
Sorry, the correct answer is 95122

=== Code Execution Successful ===
```

3.11 (Find the number of days in a month) Write a program that prompts the user to enter the month and year and displays the number of days in the month. For example, if the user entered month 2 and year 2012, the program should display that February 2012 has 29 days. If the user entered month 3 and year 2015, the program should display that March 2015 has 31 days.

```
import java.util.Scanner;


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


    System.out.print("Enter the month (1-12): ");
    int month = scanner.nextInt();


    System.out.print("Enter the year: ");
    int year = scanner.nextInt();


    int daysInMonth;


    // Determine the number of days in the month
    if (month == 2) {
      // Check if the year is a leap year
      if ((year % 4 == 0 && year % 100 != 0) || year % 400 == 0) {
        daysInMonth = 29;
      } else {
```

```
    daysInMonth = 28;
  }
} else if (month == 4 || month == 6 || month == 9 || month == 11) {
  daysInMonth = 30;
} else {
  daysInMonth = 31;
}

// Display the number of days in the month
String monthName;
switch (month) {
  case 1: monthName = "January"; break;
  case 2: monthName = "February"; break;
  case 3: monthName = "March"; break;
  case 4: monthName = "April"; break;
  case 5: monthName = "May"; break;
  case 6: monthName = "June"; break;
  case 7: monthName = "July"; break;
  case 8: monthName = "August"; break;
  case 9: monthName = "September"; break;
  case 10: monthName = "October"; break;
  case 11: monthName = "November"; break;
  case 12: monthName = "December"; break;
  default: monthName = "Invalid month"; break;
```

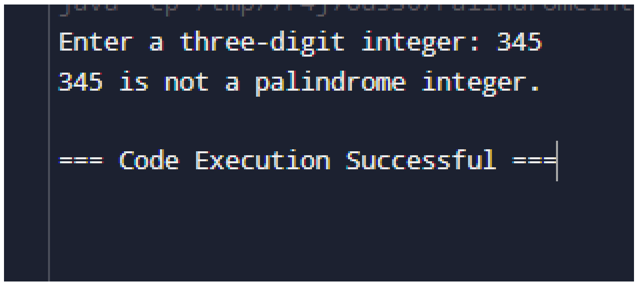
```
    }

    System.out.println(monthName + " " + year + " has " + daysInMonth +
  " days.");
  }
}
```



```
Enter the month (1-12): 5
Enter the year: 2024
May 2024 has 31 days.

=== Code Execution Successful ===
```

3.12 (Palindrome integer) Write a program that prompts the user to enter a three-digit integer and determines whether it is a palindrome integer. An integer is palindrome if it reads the same from right to left and from left to right. A negative integer is treated the same as a positive integer.

```
import java.util.Scanner;

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

    System.out.print("Enter a three-digit integer: ");
    int num = scanner.nextInt();

    // Convert the integer to a string to easily access its digits
    String strNum = String.valueOf(Math.abs(num)); // ignore sign

    // Check if the integer is a palindrome
    if (strNum.length() != 3) {
      System.out.println("Error: Input must be a three-digit integer.");
    } else if (strNum.charAt(0) == strNum.charAt(2)) {
      System.out.println(num + " is a palindrome integer.");
    } else {
      System.out.println(num + " is not a palindrome integer.");
    }
  }
}
```



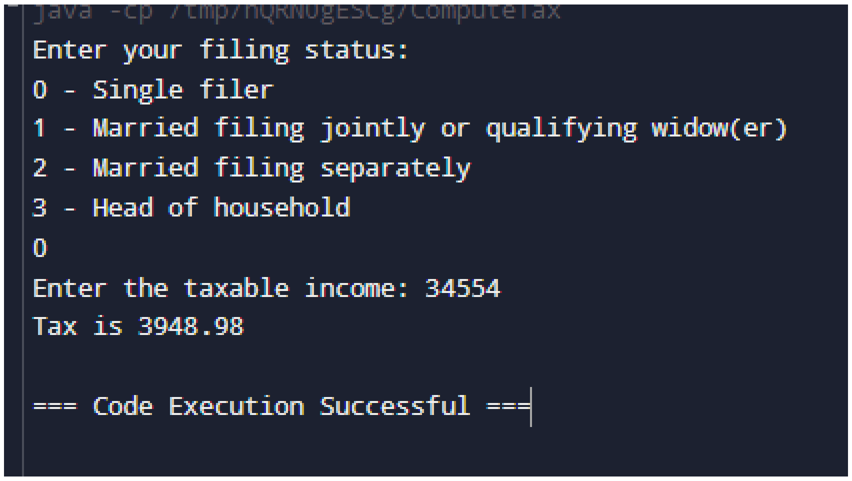
```
Enter a three-digit integer: 345
345 is not a palindrome integer.

=== Code Execution Successful ===
```

3.13 (Financial application: compute taxes) Listing 3.5, ComputeTax.java, gives the source code to compute taxes for single filers. Complete this program to compute taxes for all filing statuses.

```
import java.util.Scanner;


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


    // Prompt the user to enter filing status
    System.out.println("Enter your filing status:");
    System.out.println("0 - Single filer");
    System.out.println("1 - Married filing jointly or qualifying
widow(er)");
    System.out.println("2 - Married filing separately");
    System.out.println("3 - Head of household");
    int status = input.nextInt();


    // Prompt the user to enter taxable income
    System.out.print("Enter the taxable income: ");
    double income = input.nextDouble();


    double tax = 0;


    if (status == 0) { // Compute tax for single filers
```

```
            if (income <= 9875)
                tax = income * 0.10;
            else if (income <= 40125)
                tax = 9875 * 0.10 + (income - 9875) * 0.12;
            else if (income <= 85525)
                tax = 9875 * 0.10 + (40125 - 9875) * 0.12 + (income - 40125) *
0.22;
            // Add other brackets as necessary...
        } else if (status == 1) { // Compute tax for married filing jointly
            if (income <= 19750)
                tax = income * 0.10;
            else if (income <= 80250)
                tax = 19750 * 0.10 + (income - 19750) * 0.12;
            else if (income <= 171050)
                tax = 19750 * 0.10 + (80250 - 19750) * 0.12 + (income - 80250) *
0.22;
            // Add other brackets as necessary...
        } else if (status == 2) { // Compute tax for married filing separately
            if (income <= 9875)
                tax = income * 0.10;
            else if (income <= 40125)
                tax = 9875 * 0.10 + (income - 9875) * 0.12;
            else if (income <= 85525)
                tax = 9875 * 0.10 + (40125 - 9875) * 0.12 + (income - 40125) *
0.22;
```

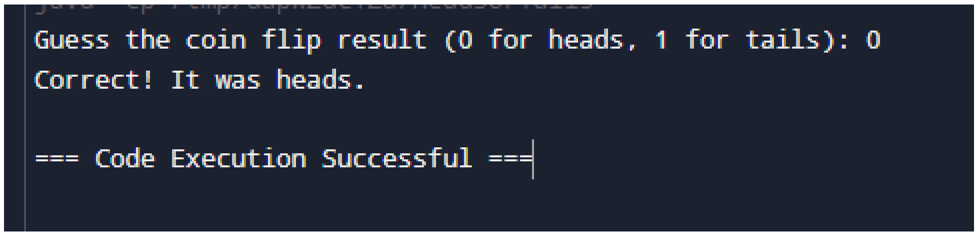
```
        // Add other brackets as necessary...
      } else if (status == 3) { // Compute tax for head of household
        if (income <= 14100)
          tax = income * 0.10;
        else if (income <= 53700)
          tax = 14100 * 0.10 + (income - 14100) * 0.12;
        else if (income <= 85500)
          tax = 14100 * 0.10 + (53700 - 14100) * 0.12 + (income - 53700) *
0.22;
        // Add other brackets as necessary...
      } else {
        System.out.println("Error: invalid status");
        System.exit(1);
      }

      // Display the result
      System.out.println("Tax is " + (int)(tax * 100) / 100.0);
    }
}
```



```
java -cp /tmp/hQRN0gESCg/ComputeTax
Enter your filing status:
0 - Single filer
1 - Married filing jointly or qualifying widow(er)
2 - Married filing separately
3 - Head of household
0
Enter the taxable income: 34554
Tax is 3948.98

=== Code Execution Successful ===
```

3.14 (Game: heads or tails) Write a program that lets the user guess whether the flip of
a coin results in heads or tails. The program randomly generates an integer 0 or 1, which represents head or tail. The program prompts the user to enter a guess, and reports whether the guess is correct or incorrect.

```
import java.util.Scanner;


public class HeadsOrTails {
    public static void main(String[] args) {
        // Create a Scanner object for user input
        Scanner input = new Scanner(System.in);


        // Randomly generate 0 or 1
        int coinFlip = (int) (Math.random() * 2);


        // Prompt the user to enter their guess
        System.out.print("Guess the coin flip result (0 for heads, 1 for tails): ");

        int guess = input.nextInt();


        // Check if the guess is correct

        if (guess == coinFlip) {

            System.out.println("Correct! It was " + (coinFlip == 0 ? "heads" : "tails") + ".");

        } else {

            System.out.println("Incorrect. It was " + (coinFlip == 0 ? "heads" : "tails") + ".");

        }

    }

}
```



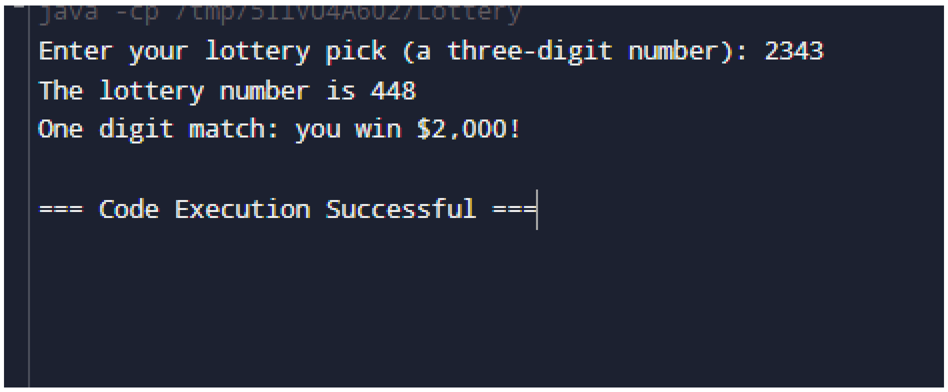
```
Guess the coin flip result (0 for heads, 1 for tails): 0
Correct! It was heads.

=== Code Execution Successful ===
```

3.15 (Game: lottery) Revise Listing 3.8, Lottery.java, to generate a lottery of a three-digit number. The program prompts the user to enter a three-digit number and determines whether the user wins according to the following rules:

1. If the user input matches the lottery number in the exact order, the award is $12,000.
2. If all digits in the user input match all digits in the lottery number, the award is $5,000.

3. If one digit in the user input matches a digit in the lottery number, the award is $2,000.

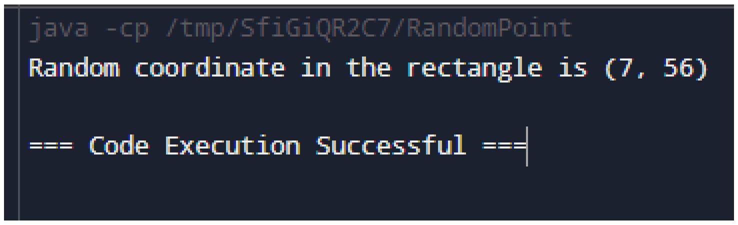
```
import java.util.Scanner;


public class Lottery {
    public static void main(String[] args) {
        // Generate a random three-digit lottery number
        int lottery = (int)(Math.random() * 1000);


        // Prompt the user to enter a three-digit number
        Scanner input = new Scanner(System.in);
        System.out.print("Enter your lottery pick (a three-digit number): ");
        int guess = input.nextInt();


        // Extract digits from lottery number
        int lotteryDigit1 = lottery / 100;
        int lotteryDigit2 = (lottery / 10) % 10;
        int lotteryDigit3 = lottery % 10;


        // Extract digits from guess
        int guessDigit1 = guess / 100;
        int guessDigit2 = (guess / 10) % 10;
        int guessDigit3 = guess % 10;
    System.out.println("The lottery number is " + lottery);


    // Check the guess against the lottery number

    if (guess == lottery) {

      System.out.println("Exact match: you win $12,000!");

    } else if ((guessDigit1 == lotteryDigit1 || guessDigit1 == lotteryDigit2
|| guessDigit1 == lotteryDigit3) &&

          (guessDigit2 == lotteryDigit1 || guessDigit2 == lotteryDigit2 ||
guessDigit2 == lotteryDigit3) &&

          (guessDigit3 == lotteryDigit1 || guessDigit3 == lotteryDigit2 ||
guessDigit3 == lotteryDigit3)) {

      System.out.println("All digits match: you win $5,000!");

    } else if (guessDigit1 == lotteryDigit1 || guessDigit1 == lotteryDigit2
|| guessDigit1 == lotteryDigit3 ||

          guessDigit2 == lotteryDigit1 || guessDigit2 == lotteryDigit2 ||
guessDigit2 == lotteryDigit3 ||

          guessDigit3 == lotteryDigit1 || guessDigit3 == lotteryDigit2 ||
guessDigit3 == lotteryDigit3) {

      System.out.println("One digit match: you win $2,000!");

    } else {

      System.out.println("Sorry, no match.");

    }

  }

}
```



```
java -cp /tmp/5l1vU4A6U2/Lottery
Enter your lottery pick (a three-digit number): 2343
The lottery number is 448
One digit match: you win $2,000!

=== Code Execution Successful ===
```

3.16 (Random point) Write a program that displays a random coordinate in a rectangle. The rectangle is centred at (0, 0) with width 50 and height 150.

```
public class RandomPoint {
  public static void main(String[] args) {
    // Generate a random x-coordinate between -25 and 25
    int x = (int)(Math.random() * 51) - 25;

    // Generate a random y-coordinate between -75 and 75
    int y = (int)(Math.random() * 151) - 75;

    // Display the random coordinate
    System.out.println("Random coordinate in the rectangle is (" + x + ", " + y + ")");
  }
}
```

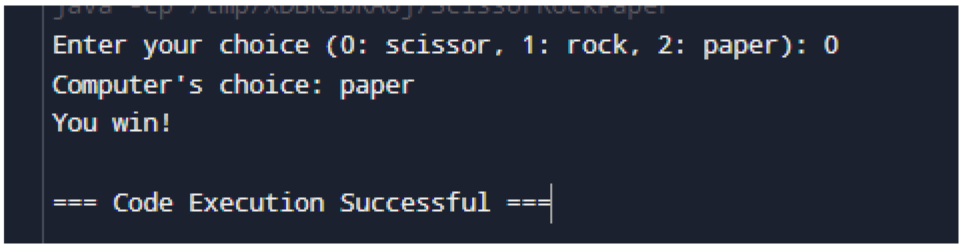


```
java -cp /tmp/SfiGiQR2C7/RandomPoint
Random coordinate in the rectangle is (7, 56)

=== Code Execution Successful ===
```

3.17 (Game: scissor, rock, paper) Write a program that plays the popular scissor–rock–paper game. (A scissor can cut a paper, a rock can knock a scissor, and a paper can wrap a rock.) The program randomly generates a number 0, 1, or 2 representing scissor, rock, and paper. The program prompts the user to enter a number 0, 1, or 2 and displays a message indicating whether the user or the computer wins, loses, or draws.

```
import java.util.Scanner;
import java.util.Random;


public class ScissorRockPaper {
  public static void main(String[] args) {
    Scanner scanner = new Scanner(System.in);
    Random random = new Random();


    int computerChoice = random.nextInt(3); // 0: scissor, 1: rock, 2: paper


    System.out.print("Enter your choice (0: scissor, 1: rock, 2: paper): ");
    int userChoice = scanner.nextInt();
```

```
String computerChoiceString;
switch (computerChoice) {
  case 0:
    computerChoiceString = "scissor";
    break;
  case 1:
    computerChoiceString = "rock";
    break;
  case 2:
    computerChoiceString = "paper";
    break;
  default:
    computerChoiceString = "";
}

String userChoiceString;
switch (userChoice) {
  case 0:
    userChoiceString = "scissor";
    break;
  case 1:
    userChoiceString = "rock";
    break;
  case 2:
        userChoiceString = "paper";
        break;
      default:
        userChoiceString = "";
    }

    System.out.println("Computer's choice: " + computerChoiceString);

    if (userChoice == computerChoice) {
      System.out.println("It's a draw!");
    } else if ((userChoice == 0 && computerChoice == 2) || // scissor cuts
paper
            (userChoice == 1 && computerChoice == 0) || // rock knocks
scissor
            (userChoice == 2 && computerChoice == 1)) { // paper wraps
rock
      System.out.println("You win!");
    } else {
      System.out.println("You lose!");
    }
  }
}
```

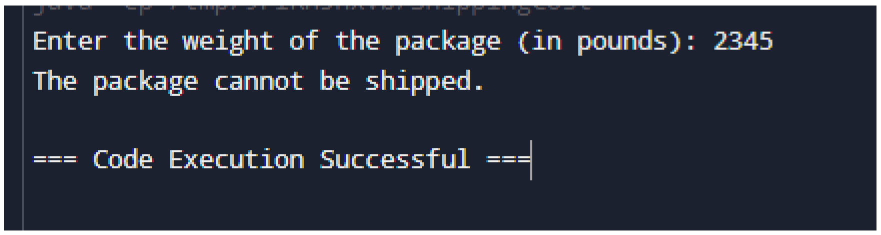

```
Enter your choice (0: scissor, 1: rock, 2: paper): 0
Computer's choice: paper
You win!

=== Code Execution Successful ===
```

3.18 (Cost of shipping) A shipping company uses the following function to calculate the cost (in dollars) of shipping based on the weight of the package (in pounds).

c(w) = d
2.5, if 0 6 w 6 = 2
4.5, if 2 6 w 6 = 4
7.5, if 4 6 w 6 = 10
10.5, if 10 6 w 6 = 20

Write a program that prompts the user to enter the weight of the package and display the shipping cost. If the weight is greater than 20, display a message “the package cannot be shipped.”

```
import java.util.Scanner;


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


    System.out.print("Enter the weight of the package (in pounds): ");
    double weight = scanner.nextDouble();
    double shippingCost;

    if (weight <= 2) {
      shippingCost = 2.5;
    } else if (weight <= 4) {
      shippingCost = 4.5;
    } else if (weight <= 10) {
      shippingCost = 7.5;
    } else if (weight <= 20) {
      shippingCost = 10.5;
    } else {
      System.out.println("The package cannot be shipped.");
      return;
    }

    System.out.println("The shipping cost is $" + shippingCost);
  }
}
```

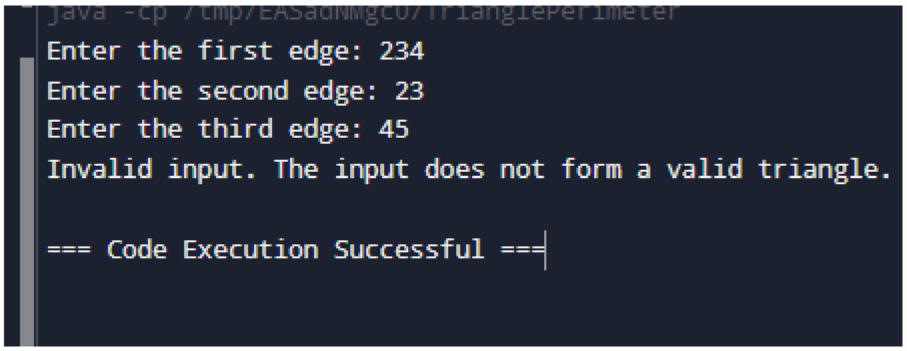


```
Enter the weight of the package (in pounds): 2345
The package cannot be shipped.

=== Code Execution Successful ===
```

3.19 (Compute the perimeter of a triangle) Write a program that reads three edges for a triangle and computes the perimeter if the input is valid. Otherwise, display that the input is invalid. The input is valid if the sum of every pair of two edges is greater than the remaining edge.

```
import java.util.Scanner;

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

    System.out.print("Enter the first edge: ");
    double edge1 = scanner.nextDouble();

    System.out.print("Enter the second edge: ");
    double edge2 = scanner.nextDouble();

    System.out.print("Enter the third edge: ");
    double edge3 = scanner.nextDouble();

    if (isValidInput(edge1, edge2, edge3)) {
      double perimeter = edge1 + edge2 + edge3;
      System.out.println("The perimeter of the triangle is " + perimeter);
    } else {
      System.out.println("Invalid input. The input does not form a valid
triangle.");
```

```
   }
  }

  public static boolean isValidInput(double edge1, double edge2,
 double edge3) {
    return (edge1 + edge2 > edge3) && (edge1 + edge3 > edge2) && (edge2
 + edge3 > edge1);
  }
 }
```



```
java -cp /tmp/EASadNMgCO/TrianglePerimeter
Enter the first edge: 234
Enter the second edge: 23
Enter the third edge: 45
Invalid input. The input does not form a valid triangle.

=== Code Execution Successful ===
```

3.20 (Science: wind-chill temperature) Programming Exercise 2.17 gives a formula to compute the wind-chill temperature. The formula is valid for temperatures in the range between -58°F and 41°F and wind speed greater than or equal to 2. Write a program that prompts the user to enter a temperature and a wind speed. The program displays the wind-chill temperature if the input is valid; otherwise, it displays a message indicating whether the temperature and/or wind speed is invalid.

```
 package ch_03;
```

```
import java.util.*;


public class Exercise03_20 extends Exercise03_21 {
  public static void main(String[] args) {


    Scanner input = new Scanner(System.in);
    System.out.println("Enter the temperature in Fahrenheit between -
58°F and 41°F: ");
    double temp = input.nextDouble();


    System.out.println("Enter the wind speed (>=2) in miles per hour:
");
    int windSpeed = input.nextInt();


    if (temp >= -58 && temp <= 41 && windSpeed >= 2) {


      double vToPow16 = Math.pow(windSpeed, 0.16);


      double twc = 35.74 + 0.6215 * temp - 35.75 * vToPow16 + 0.4275
* temp * vToPow16;
      System.out.println("The wind chill index is " + twc);
    } else {
      System.out.println("Please check to make sure you are entering
valid input.");
```

```
        }
        input.close();
    }
 }
```

3.21 (Science: day of the week) Zeller's congruence is an algorithm developed by Christian Zeller to calculate the day of the week. The formula is

$$h = (q + \frac{26(m + 1)}{10} + k + \frac{k}{4} + \frac{j}{4} + 5j) \% 7$$

where

■ h is the day of the week (0: Saturday, 1: Sunday, 2: Monday, 3: Tuesday, 4: Wednesday, 5: Thursday, and 6: Friday).

■ q is the day of the month.

■ m is the month (3: March, 4: April, . . ., 12: December). January and February are counted as months 13 and 14 of the previous year.

■ j is year 100.

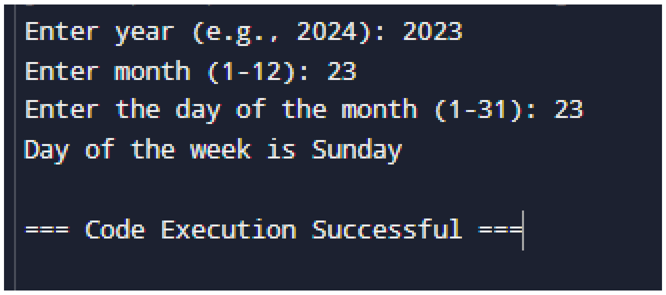
■ k is the year of the century (i.e., year % 100).

Note all divisions in this exercise perform an integer division. Write a program that prompts the user to enter a year, month, and day of the month, and displays the name of the day of the week.

```
import java.util.Scanner;


public class ZellersCongruence {
    public static void main(String[] args) {
```

```
Scanner input = new Scanner(System.in);

// Prompt the user to enter year, month, and day
System.out.print("Enter year (e.g., 2024): ");
int year = input.nextInt();

System.out.print("Enter month (1-12): ");
int month = input.nextInt();

System.out.print("Enter the day of the month (1-31): ");
int day = input.nextInt();

// Adjust month and year for January and February
if (month == 1 || month == 2) {
  month += 12;
  year -= 1;
}

// Calculate k and j
int k = year % 100;    // year of the century
int j = year / 100;    // century

// Apply Zeller's congruence formula
int h = (day + (26 * (month + 1)) / 10 + k + k / 4 + j / 4 + 5 * j) % 7;
```

```
        // Determine the day of the week
        String dayOfWeek = "";
        switch (h) {
            case 0: dayOfWeek = "Saturday"; break;
            case 1: dayOfWeek = "Sunday"; break;
            case 2: dayOfWeek = "Monday"; break;
            case 3: dayOfWeek = "Tuesday"; break;
            case 4: dayOfWeek = "Wednesday"; break;
            case 5: dayOfWeek = "Thursday"; break;
            case 6: dayOfWeek = "Friday"; break;
        }

        // Display the result
        System.out.println("Day of the week is " + dayOfWeek);
    }
}
```

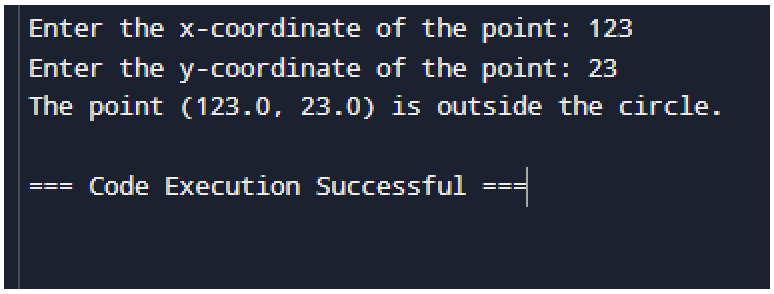


```
Enter year (e.g., 2024): 2023
Enter month (1-12): 23
Enter the day of the month (1-31): 23
Day of the week is Sunday

=== Code Execution Successful ===
```

3.22 (Geometry: point in a circle?) Write a program that prompts the user to enter a point (x, y) and checks whether the point is

within the circle centered at (0, 0) with radius 10. For example, (4, 5) is inside the circle and (9, 9) is outside the circle, as shown in Figure 3.7a. (Hint: A point is in the circle if its distance to (0, 0) is less than or equal to 10.
The formula for computing the distance is $\sqrt{((x2 - x1)^2 + (y2 - y1)^2)}$. Test your program to cover all cases.)

```
import java.util.Scanner;


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


  System.out.print("Enter the x-coordinate of the point: ");
  double x = scanner.nextDouble();


  System.out.print("Enter the y-coordinate of the point: ");
  double y = scanner.nextDouble();


  if (isPointInCircle(x, y)) {
   System.out.println("The point (" + x + ", " + y + ") is inside the circle.");
  } else {
   System.out.println("The point (" + x + ", " + y + ") is outside the circle.");
  }
  }

  public static boolean isPointInCircle(double x, double y) {
    double distance = Math.sqrt(Math.pow(x, 2) + Math.pow(y, 2));
    return distance <= 10;
  }
}
```

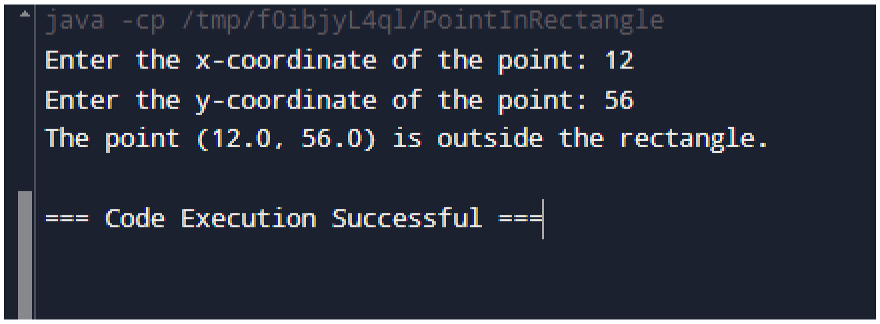


```
Enter the x-coordinate of the point: 123
Enter the y-coordinate of the point: 23
The point (123.0, 23.0) is outside the circle.

=== Code Execution Successful ===
```

3.23 (Geometry: point in a rectangle?) Write a program that prompts the user to enter a point (x, y) and checks whether the point is within the rectangle centred at (1, 1) with width 10 and height 5. For example, (2, 2) is inside the rectangle and (6, 4) is outside the rectangle, as shown in Figure 3.7b. (Hint: A point is in the rectangle if its horizontal distance to (0, 0) is less than or equal to 10 / 2 and its vertical distance to (0, 0) is less than or equal to 5.0 / 2. Test your program to cover all cases.)

```
import java.util.Scanner;

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


  System.out.print("Enter the x-coordinate of the point: ");
  double x = scanner.nextDouble();


  System.out.print("Enter the y-coordinate of the point: ");
  double y = scanner.nextDouble();


  if (isPointInRectangle(x, y)) {
   System.out.println("The point (" + x + ", " + y + ") is inside the
rectangle.");
  } else {
   System.out.println("The point (" + x + ", " + y + ") is outside the
rectangle.");
  }
 }


 public static boolean isPointInRectangle(double x, double y) {
  double centerX = 1;
  double centerY = 1;
  double width = 10;
  double height = 5;


  double horizontalDistance = Math.abs(x - centerX);
```

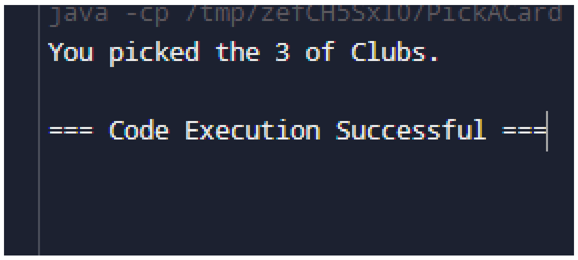
```
    double verticalDistance = Math.abs(y - centerY);


    return horizontalDistance <= width / 2 && verticalDistance <= height / 2;
  }
}
```



```
java -cp /tmp/f0ibjyL4ql/PointInRectangle
Enter the x-coordinate of the point: 12
Enter the y-coordinate of the point: 56
The point (12.0, 56.0) is outside the rectangle.

=== Code Execution Successful ===
```

3.24 (Game: pick a card) Write a program that simulates picking a card from a deck of 52 cards. Your program should display the rank (Ace, 2, 3, 4, 5, 6, 7, 8, 9, 10, Jack, Queen, King) and suit (Clubs, Diamonds, Hearts, Spades) of the card.

```
import java.util.Random;


public class PickACard {
  public static void main(String[] args) {
    String[] ranks = {"Ace", "2", "3", "4", "5", "6", "7", "8", "9", "10", "Jack", "Queen", "King"};
    String[] suits = {"Clubs", "Diamonds", "Hearts", "Spades"};


    Random random = new Random();
    int rankIndex = random.nextInt(ranks.length);
    int suitIndex = random.nextInt(suits.length);

    String rank = ranks[rankIndex];
    String suit = suits[suitIndex];

    System.out.println("You picked the " + rank + " of " + suit + ".");
  }
}
```

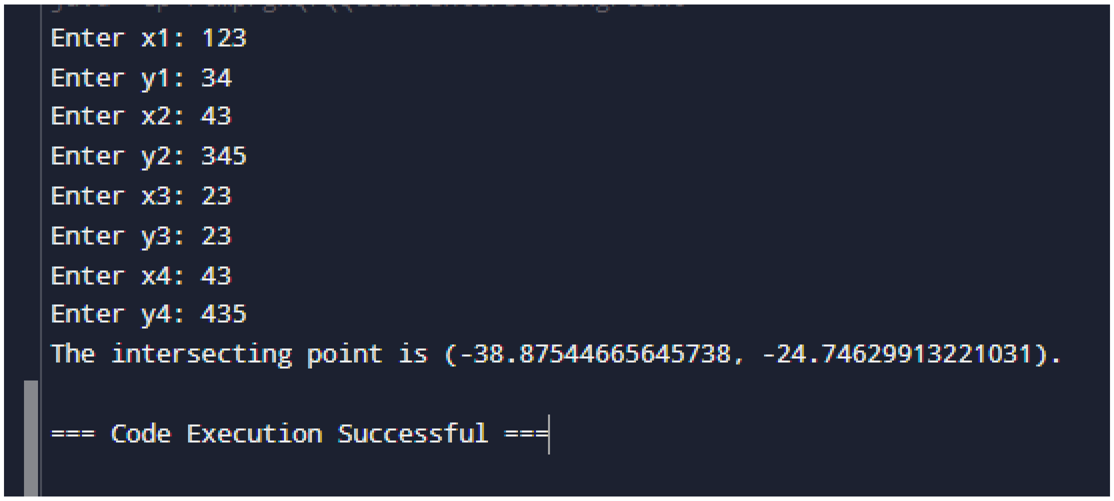


```
java -cp /tmp/zefCH5Sx10/PickACard
You picked the 3 of Clubs.

=== Code Execution Successful ===
```

3.25 (Geometry: intersecting point) Two points on line 1 are given as (x_1, y_1) and (x_2, y_2) and on line 2 as (x_3, y_3) and (x_4, y_4), as shown in Figure 3.8a and b. The intersecting point of the two lines can be found by solving the following linear equations:
$(y_1 - y_2)x - (x_1 - x_2)y = (y_1 - y_2)x_1 - (x_1 - x_2)y_1$
$(y_3 - y_4)x - (x_3 - x_4)y = (y_3 - y_4)x_3 - (x_3 - x_4)y_3$
This linear equation can be solved using Cramer’s rule (see Programming Exercise 3.3). If the equation has no solutions, the two lines are parallel (see Figure 3.8c). Write a program that

prompts the user to enter four points and displays the intersecting point.

```
import java.util.Scanner;


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


    System.out.print("Enter x1: ");
    double x1 = scanner.nextDouble();


    System.out.print("Enter y1: ");
    double y1 = scanner.nextDouble();


    System.out.print("Enter x2: ");
    double x2 = scanner.nextDouble();


    System.out.print("Enter y2: ");
    double y2 = scanner.nextDouble();


    System.out.print("Enter x3: ");
    double x3 = scanner.nextDouble();


    System.out.print("Enter y3: ");
    double y3 = scanner.nextDouble();

    System.out.print("Enter x4: ");
    double x4 = scanner.nextDouble();

    System.out.print("Enter y4: ");
    double y4 = scanner.nextDouble();

    double denominator = (x1 - x2) * (y3 - y4) - (y1 - y2) * (x3 - x4);

    if (denominator == 0) {
      System.out.println("The two lines are parallel and do not intersect.");
    } else {
      double x = ((y1 - y2) * (x3 - x4) * x1 - (x1 - x2) * (y3 - y4) * x3) /
denominator;
      double y = ((y1 - y2) * (x3 - x4) * y1 - (x1 - x2) * (y3 - y4) * y3) /
denominator;

      System.out.println("The intersecting point is (" + x + ", " + y + ").");
    }
  }
}
```


```
Enter x1: 123
Enter y1: 34
Enter x2: 43
Enter y2: 345
Enter x3: 23
Enter y3: 23
Enter x4: 43
Enter y4: 435
The intersecting point is (-38.87544665645738, -24.74629913221031).

=== Code Execution Successful ===
```

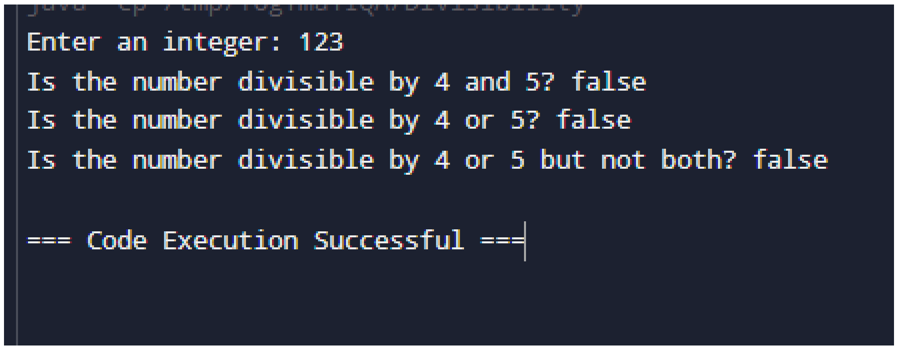
3.26 (Use the &&, || and ^ operators) Write a program that prompts the user to enter an integer and determines whether it is divisible by 4 and 5, whether it is divisible by 4 or 5, and whether it is divisible by 4 or 5 but not both.

```
package ch_03;


import java.util.*;


/**
 * 3.26 (Use the &&, || and ^ operators)
 * Write a program that prompts the user to enter
 * an integer and determines whether it is divisible by 5 and 6, whether it is divisible
 * by 5 or 6, and whether it is divisible by 5 or 6, but not both. Here is a sample run
 * of this program:
 * <p>
 * Enter an integer: 10
 * Is 10 divisible by 5 and 6? false
 * Is 10 divisible by 5 or 6? true
 * Is 10 divisible by 5 or 6, but not both? true
 */
public class Exercise03_26 {
   public static void main(String[] args) {
      Scanner in = new Scanner(System.in);
      System.out.println("Enter an integer: ");
      int userInt = in.nextInt();

      boolean fiveAndSix = userInt % 5 == 0 && userInt % 6 == 0;
      boolean fiveOrSix = userInt % 5 == 0 || userInt % 6 == 0;
      boolean fiveSixNotBoth = (userInt % 5 == 0 || userInt % 6 == 0) ^
(userInt % 5 == 0 && userInt % 6 == 0);

      System.out.println("Is " + userInt + " divisible by 5 and 6? " +
fiveAndSix);
      System.out.println("Is " + userInt + " divisible by 5 or 6? " +
fiveOrSix);
      System.out.println("Is " + userInt + " divisible by 5 or 6, but not both?
" + fiveSixNotBoth);



   }
}
```

3.27 (Geometry: points in triangle?) Suppose a right triangle is placed in a plane as shown below. The right-angle point is placed at (0, 0), and the other two points are placed at (200, 0) and (0, 100). Write a program that prompts the user to enter a point with x- and y-coordinates and determines whether the point is inside the triangle.

```
import java.util.Scanner;


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


    System.out.print("Enter an integer: ");
    int num = scanner.nextInt();


    boolean isDivisibleBy4And5 = (num % 4 == 0) && (num % 5 == 0);
    boolean isDivisibleBy4Or5 = (num % 4 == 0) || (num % 5 == 0);
    boolean isDivisibleBy4Or5ButNotBoth = (num % 4 == 0) ^ (num % 5
== 0);


    System.out.println("Is the number divisible by 4 and 5? " +
isDivisibleBy4And5);
    System.out.println("Is the number divisible by 4 or 5? " +
isDivisibleBy4Or5);
```

```
    System.out.println("Is the number divisible by 4 or 5 but not both? " +
isDivisibleBy4Or5ButNotBoth);
  }
}
```



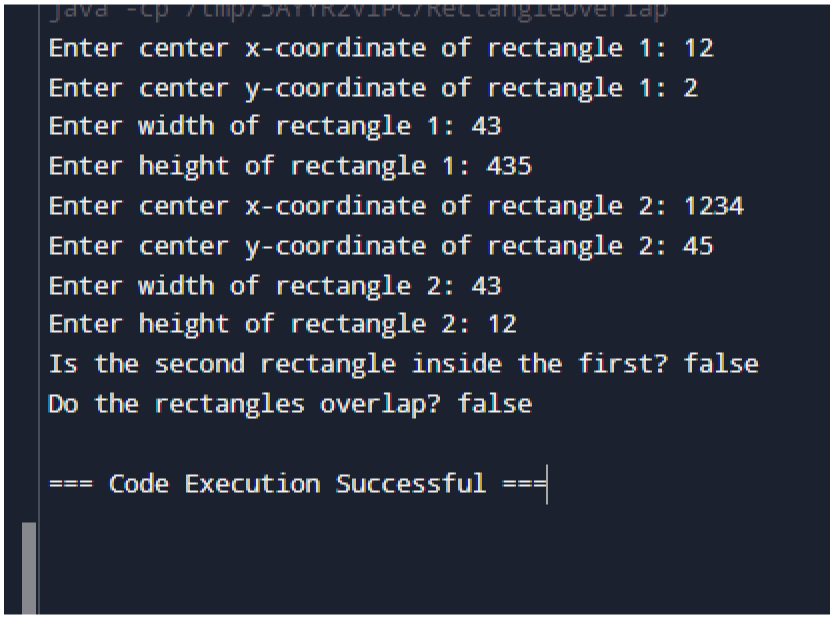
```
Enter an integer: 123
Is the number divisible by 4 and 5? false
Is the number divisible by 4 or 5? false
Is the number divisible by 4 or 5 but not both? false

=== Code Execution Successful ===
```

3.28 (Geometry: two rectangles) Write a program that prompts the user to enter the center x-, y-coordinates, width, and height of two rectangles and determines whether the second rectangle is inside the first or overlaps with the first, as shown in Figure 3.9. Test your program to cover all cases.

```
import java.util.Scanner;

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

    // Get the coordinates and dimensions of the first rectangle
    System.out.print("Enter center x-coordinate of rectangle 1: ");
```

```
double x1 = scanner.nextDouble();

System.out.print("Enter center y-coordinate of rectangle 1: ");
double y1 = scanner.nextDouble();

System.out.print("Enter width of rectangle 1: ");
double w1 = scanner.nextDouble();

System.out.print("Enter height of rectangle 1: ");
double h1 = scanner.nextDouble();

// Get the coordinates and dimensions of the second rectangle
System.out.print("Enter center x-coordinate of rectangle 2: ");
double x2 = scanner.nextDouble();

System.out.print("Enter center y-coordinate of rectangle 2: ");
double y2 = scanner.nextDouble();

System.out.print("Enter width of rectangle 2: ");
double w2 = scanner.nextDouble();

System.out.print("Enter height of rectangle 2: ");
double h2 = scanner.nextDouble();
```

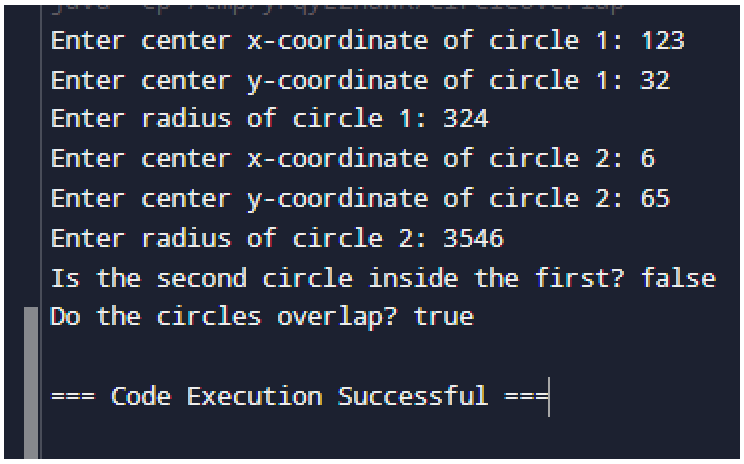
```
   // Calculate the boundaries of the rectangles
   double left1 = x1 - w1 / 2;
   double right1 = x1 + w1 / 2;
   double top1 = y1 + h1 / 2;
   double bottom1 = y1 - h1 / 2;

   double left2 = x2 - w2 / 2;
   double right2 = x2 + w2 / 2;
   double top2 = y2 + h2 / 2;
   double bottom2 = y2 - h2 / 2;

   // Check if the second rectangle is inside the first
   boolean isInside = (left1 <= left2) && (right1 >= right2) && (top1 >=
top2) && (bottom1 <= bottom2);

   // Check if the second rectangle overlaps with the first
   boolean doesOverlap = !(right1 < left2 || left1 > right2 || top1 <
bottom2 || bottom1 > top2);

   System.out.println("Is the second rectangle inside the first? " +
isInside);
   System.out.println("Do the rectangles overlap? " + doesOverlap);
  }
}
```



```
Enter center x-coordinate of rectangle 1: 12
Enter center y-coordinate of rectangle 1: 2
Enter width of rectangle 1: 43
Enter height of rectangle 1: 435
Enter center x-coordinate of rectangle 2: 1234
Enter center y-coordinate of rectangle 2: 45
Enter width of rectangle 2: 43
Enter height of rectangle 2: 12
Is the second rectangle inside the first? false
Do the rectangles overlap? false

=== Code Execution Successful ===
```

3.29 (Geometry: two circles) Write a program that prompts the user to enter the center coordinates and radii of two circles and determines whether the second circle is inside the first or overlaps with the first, as shown in Figure 3.10. (Hint: circle2 is inside circle1 if the distance between the two centers 6 = r1 - r2 and circle2 overlaps circle1 if the distance between the two centers 6 = r1 + r2. Test your program to cover all cases.)

```
import java.util.Scanner;


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


    // Get the center coordinates and radii of the first circle
    System.out.print("Enter center x-coordinate of circle 1: ");
    double x1 = scanner.nextDouble();

    System.out.print("Enter center y-coordinate of circle 1: ");
    double y1 = scanner.nextDouble();

    System.out.print("Enter radius of circle 1: ");
    double r1 = scanner.nextDouble();

    // Get the center coordinates and radii of the second circle
    System.out.print("Enter center x-coordinate of circle 2: ");
    double x2 = scanner.nextDouble();

    System.out.print("Enter center y-coordinate of circle 2: ");
    double y2 = scanner.nextDouble();

    System.out.print("Enter radius of circle 2: ");
    double r2 = scanner.nextDouble();

    // Calculate the distance between the two centers
    double distance = Math.sqrt(Math.pow(x2 - x1, 2) + Math.pow(y2 - y1,
2));

    // Check if the second circle is inside the first
    boolean isInside = distance <= r1 - r2;
    // Check if the second circle overlaps with the first
    boolean doesOverlap = distance <= r1 + r2;

    System.out.println("Is the second circle inside the first? " + isInside);
    System.out.println("Do the circles overlap? " + doesOverlap);
  }
}
```

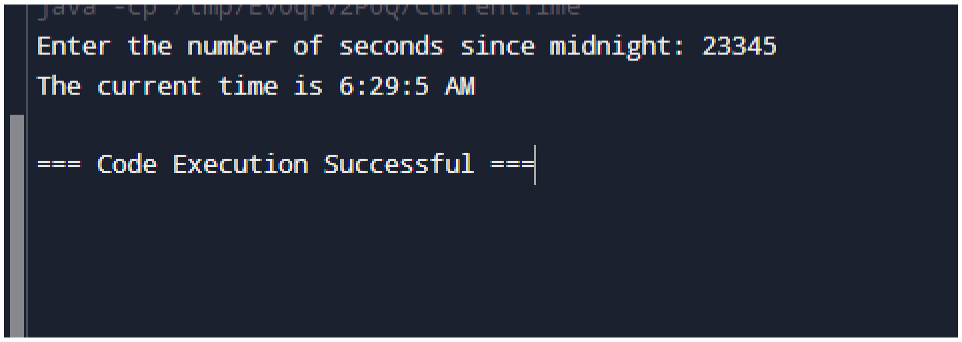


```
Enter center x-coordinate of circle 1: 123
Enter center y-coordinate of circle 1: 32
Enter radius of circle 1: 324
Enter center x-coordinate of circle 2: 6
Enter center y-coordinate of circle 2: 65
Enter radius of circle 2: 3546
Is the second circle inside the first? false
Do the circles overlap? true

=== Code Execution Successful ===
```

3.30 (Current time) Revise Programming Exercise 2.8 to display the hour using a 12-hour clock.

```
import java.util.Scanner;

public class CurrentTime {
  public static void main(String[] args) {
    Scanner scanner = new Scanner(System.in);
```

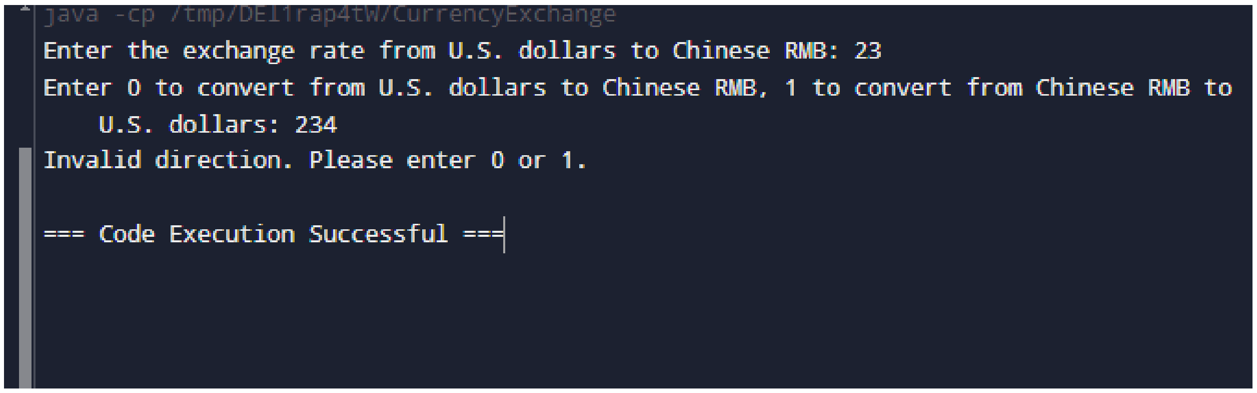
```
// Get the current time in seconds
System.out.print("Enter the number of seconds since midnight: ");
int totalSeconds = scanner.nextInt();

// Calculate the hour, minute, and second
int seconds = totalSeconds % 60;
int totalMinutes = totalSeconds / 60;
int minutes = totalMinutes % 60;
int hours = totalMinutes / 60;

// Convert the hour to a 12-hour clock
String ampm;
int hour;
if (hours == 0) {
  hour = 12;
  ampm = "AM";
} else if (hours < 12) {
  hour = hours;
  ampm = "AM";
} else if (hours == 12) {
  hour = hours;
  ampm = "PM";
} else {
      hour = hours - 12;

      ampm = "PM";

    }


    // Display the current time

    System.out.println("The current time is " + hour + ":" + minutes + ":" +
seconds + " " + ampm);

  }

}
```



```
Enter the number of seconds since midnight: 23345
The current time is 6:29:5 AM

=== Code Execution Successful ===
```

3.31 (Financials: currency exchange) Write a program that prompts the user to enter the exchange rate from currency in U.S. dollars to Chinese RMB. Prompt the user to enter 0 to convert from U.S. dollars to Chinese RMB and 1 to convert from Chinese RMB to U.S. dollars. Prompt the user to enter the amount in U.S. dollars or Chinese RMB to convert it to Chinese RMB or U.S. dollars, respectively.

```
import java.util.Scanner;


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

    // Get the exchange rate from U.S. dollars to Chinese RMB
    System.out.print("Enter the exchange rate from U.S. dollars to
Chinese RMB: ");
    double exchangeRate = scanner.nextDouble();

    // Get the conversion direction
    System.out.print("Enter 0 to convert from U.S. dollars to Chinese
RMB, 1 to convert from Chinese RMB to U.S. dollars: ");
    int direction = scanner.nextInt();

    // Get the amount to convert
    if (direction == 0) {
      System.out.print("Enter the amount in U.S. dollars: ");
      double usdAmount = scanner.nextDouble();
      double rmbAmount = usdAmount * exchangeRate;
      System.out.println("The amount in Chinese RMB is " + rmbAmount);
    } else if (direction == 1) {
      System.out.print("Enter the amount in Chinese RMB: ");
      double rmbAmount = scanner.nextDouble();
      double usdAmount = rmbAmount / exchangeRate;
      System.out.println("The amount in U.S. dollars is " + usdAmount);
    } else {
      System.out.println("Invalid direction. Please enter 0 or 1.");
    }
  }
}
```

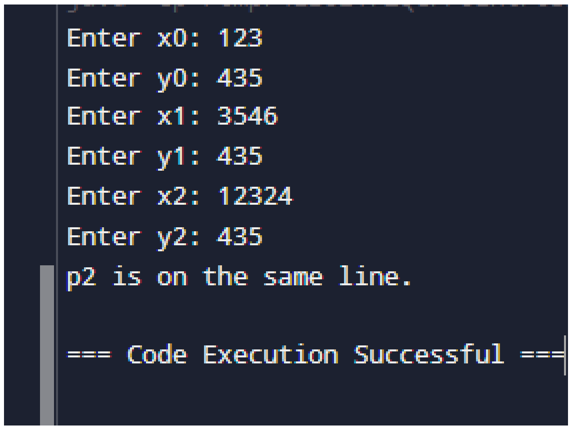


```
java -cp /tmp/DE11rap4tW/CurrencyExchange
Enter the exchange rate from U.S. dollars to Chinese RMB: 23
Enter 0 to convert from U.S. dollars to Chinese RMB, 1 to convert from Chinese RMB to
    U.S. dollars: 234
Invalid direction. Please enter 0 or 1.

=== Code Execution Successful ===
```

3.32 (Geometry: point position) Given a directed line from point p0(x0, y0) to p1(x1, y1), you can use the following condition to decide whether a point p2(x2, y2) is on the left of the line, on the right, or on the same line (see Figure 3.11)

```
import java.util.Scanner;

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

    // Get the coordinates of p0
    System.out.print("Enter x0: ");
    double x0 = scanner.nextDouble();
```

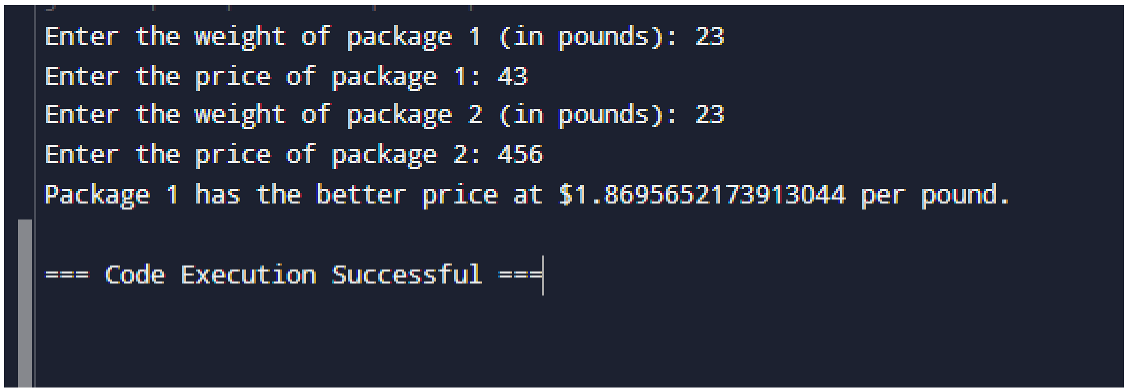
```
System.out.print("Enter y0: ");
double y0 = scanner.nextDouble();

// Get the coordinates of p1
System.out.print("Enter x1: ");
double x1 = scanner.nextDouble();
System.out.print("Enter y1: ");
double y1 = scanner.nextDouble();

// Get the coordinates of p2
System.out.print("Enter x2: ");
double x2 = scanner.nextDouble();
System.out.print("Enter y2: ");
double y2 = scanner.nextDouble();

// Calculate the position of p2
double position = (x1 - x0) * (y2 - y0) - (x2 - x0) * (y1 - y0);

// Determine the position of p2
if (position > 0) {
  System.out.println("p2 is on the left side of the line.");
} else if (position == 0) {
  System.out.println("p2 is on the same line.");
} else {
    System.out.println("p2 is on the right side of the line.");
   }
  }
 }
```



```
Enter x0: 123
Enter y0: 435
Enter x1: 3546
Enter y1: 435
Enter x2: 12324
Enter y2: 435
p2 is on the same line.

=== Code Execution Successful ===
```

3.33 (Financial: compare costs) Suppose you shop for rice in two different packages. You would like to write a program to compare the cost. The program prompts the user to enter the weight and price of each package and displays the one with the better price.

```
import java.util.Scanner;


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



    System.out.print("Enter the weight of package 1 (in pounds): ");
  double weight1 = scanner.nextDouble();
  System.out.print("Enter the price of package 1: ");
  double price1 = scanner.nextDouble();


  System.out.print("Enter the weight of package 2 (in pounds): ");
  double weight2 = scanner.nextDouble();
  System.out.print("Enter the price of package 2: ");
  double price2 = scanner.nextDouble();


  double costPerPound1 = price1 / weight1;
  double costPerPound2 = price2 / weight2;


  if (costPerPound1 < costPerPound2) {
   System.out.println("Package 1 has the better price at $" +
costPerPound1 + " per pound.");
  } else if (costPerPound1 > costPerPound2) {
   System.out.println("Package 2 has the better price at $" +
costPerPound2 + " per pound.");
  } else {
   System.out.println("Both packages have the same price at $" +
costPerPound1 + " per pound.");
  }
  }
 }
```

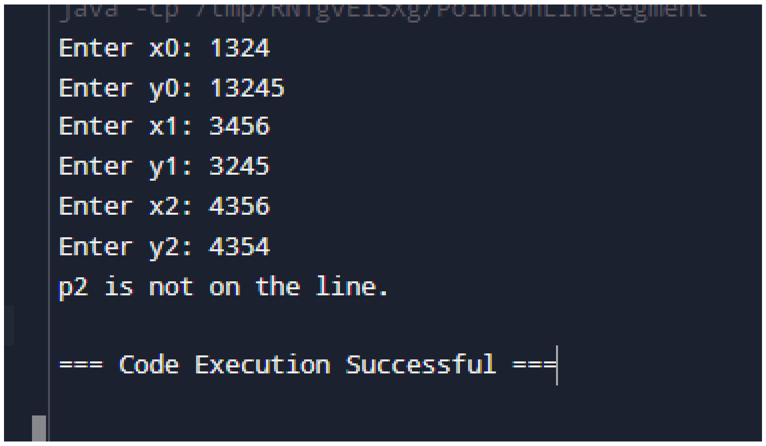


```
Enter the weight of package 1 (in pounds): 23
Enter the price of package 1: 43
Enter the weight of package 2 (in pounds): 23
Enter the price of package 2: 456
Package 1 has the better price at $1.8695652173913044 per pound.

=== Code Execution Successful ===
```

3.34 (Geometry: point on line segment) Exercise 3.32 shows how to test whether a point is on an unbounded line. Revise Exercise 3.32 to test whether a point is on a line segment. Write a program that prompts the user to enter the three points for p0, p1, and p2 and displays whether p2 is on the line segment from p0 to p1

```
import java.util.Scanner;


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



    System.out.print("Enter x0: ");
    double x0 = scanner.nextDouble();
```

```
System.out.print("Enter y0: ");
double y0 = scanner.nextDouble();



System.out.print("Enter x1: ");
double x1 = scanner.nextDouble();
System.out.print("Enter y1: ");
double y1 = scanner.nextDouble();



System.out.print("Enter x2: ");
double x2 = scanner.nextDouble();
System.out.print("Enter y2: ");
double y2 = scanner.nextDouble();

double position = (x1 - x0) * (y2 - y0) - (x2 - x0) * (y1 - y0);



if (position == 0) {

  if ((x0 <= x2 && x2 <= x1) || (x1 <= x2 && x2 <= x0)) {
    if ((y0 <= y2 && y2 <= y1) || (y1 <= y2 && y2 <= y0)) {
      System.out.println("p2 is on the line segment from p0 to p1.");
    } else {
     System.out.println("p2 is on the line but not on the line segment
from p0 to p1.");
    }
   } else {
    System.out.println("p2 is on the line but not on the line segment
from p0 to p1.");
   }
  } else {
   System.out.println("p2 is not on the line.");
  }
 }
}
```

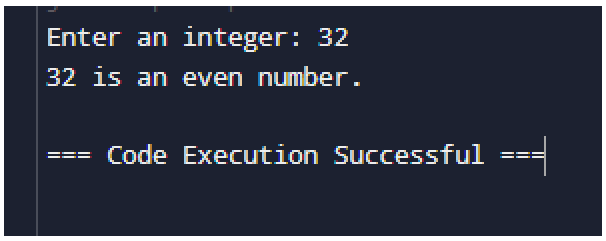


```
java -cp /tmp/RNTgvEISXg/PointOnLineSegment
Enter x0: 1324
Enter y0: 13245
Enter x1: 3456
Enter y1: 3245
Enter x2: 4356
Enter y2: 4354
p2 is not on the line.

=== Code Execution Successful ===
```

3.35 (Even or odd number) Write a program that prompts the user to enter an integer and displays whether the integer is an odd number or not.

```
import java.util.Scanner;
```

```
public class EvenOrOdd {
  public static void main(String[] args) {
    Scanner scanner = new Scanner(System.in);



    System.out.print("Enter an integer: ");
    int number = scanner.nextInt();



    if (number % 2 != 0) {
      System.out.println(number + " is an odd number.");
    } else {
      System.out.println(number + " is an even number.");
    }
  }
}
```



```
Enter an integer: 32
32 is an even number.

=== Code Execution Successful ===
```