CodInClub

Powered By **BridgeLabz**

Best Programming Practice

1. Use Variables including Fixed, User Inputs, and Results
2. Use Methods instead of writing code in the main() function
3. Proper naming conventions for all variables and methods
4. Proper Program Name and Class Name
5. Handle Checked and Unchecked Exceptions wherever possible
6. Proper Method Name which indicates action taking inputs and providing result

Sample Program 1: Create a program to find all the occurrences of a character in a string using the charAt() method

   a. Take user input for the String and occurrences of the Character to find
   b. Write a method to find all the occurrences of the characters.
      i. The logic used is to first find the number of occurrences of the character and
      ii. then create an array to store the indexes of the character
   c. Call the method in the main and display the result

```Java
// Program to find all the occurrences of a character in a string
import java.util.Scanner;
class StringAnalyzer {
  // Method to find all the index of a character in a string using charAt()
  // method and return them in an array
  public static int[] findAllIndexes(String text, char ch) {
    // The count is used to find the number of occurrences of the character
    int count = 0;
    for (int i = 0; i < text.length(); i++) {
        if (text.charAt(i) == ch) {
            count++;
        }
    }

    // Create an array to store the indexes of the character
    int[] indexes = new int[count];
    int j = 0;
    for (int i = 0; i < text.length(); i++) {
        if (text.charAt(i) == ch) {
            indexes[j] = i;
            j++;
        }
    }
    return indexes;
  }
```

```java
    public static void main(String[] args) {
        // Take user input for Text and Character to check Occurrences
        Scanner sc = new Scanner(System.in);
        System.out.print(Enter a text: ");
        String text = sc.nextLine();
        System.out.print("Enter a character to find the occurrences: ");
        char ch = sc.next().charAt(0);

        // Find the occurrences of the character
        int[] indexes = findAllIndexes(text, ch);

        // Display the occurrences of the character
        System.out.println("Indexes of the character '" + ch + "': ");
        for (int i = 0; i < indexes.length; i++) {
            System.out.print(indexes[i] + " ");
        }
    }
}
```

Level 3 Practice Programs

1. An organization took up the exercise to find the Body Mass Index (BMI) of all the persons in a team of 10 members. For this create a program to find the BMI and display the height, weight, BMI, and status of each individual

   Hint =>
   a. Take user input for the person's weight (kg) and height (cm) and store it in the corresponding 2D array of 10 rows. The First Column stores the weight and the second column stores the height in cm
   b. Create a Method to find the BMI and status of every person given the person's height and weight and return the 2D String array. Use the formula BMI = weight / (height * height). Note unit is kg/m^2. For this convert cm to meter
   c. Create a Method that takes the 2D array of height and weight as parameters. Calls the user-defined method to compute the BMI and the BMI Status and stores in a 2D String array of height, weight, BMI, and status.
   d. Create a method to display the 2D string array in a tabular format of Person's Height, Weight, BMI, and the Status
   e. Finally, the main function takes user inputs, calls the user-defined methods, and displays the result.

BMI	Status
≤ 18.4	Underweight
18.5 - 24.9	Normal
25.0 - 39.9	Overweight
≥ 40.0	Obese

```java
package Day5.LabPractice_L3;

import java.util.Scanner;

public class LP1 {

    public static String[][] computeBMI(double[][] data) {

        int numPersons = data.length;

        String[][] results = new String[numPersons][4];

        for (int i = 0; i < numPersons; i++) {

            double weight = data[i][0];

            double heightInMeters = data[i][1] / 100.0;

            double bmi = weight / (heightInMeters * heightInMeters);

            bmi = Math.round(bmi * 10.0) / 10.0;

            String status;
```

```java
            if (bmi < 18.5) {
                status = "Underweight";
            } else if (bmi >= 18.5 && bmi < 24.9) {
                status = "Normal Weight";
            } else if (bmi >= 25 && bmi < 29.9) {
                status = "Overweight";
            } else {
                status = "Obese";
            }
            results[i][0] = String.format("%.1f", data[i][1]);
            results[i][1] = String.format("%.1f", data[i][0]);
            results[i][2] = String.format("%.1f", bmi);
            results[i][3] = status;
        }
        return results;
    }

    public static void displayResults(String[][] results) {
        System.out.println("\nBMI Report:");

System.out.println("----------------------------------------------------
-");
        System.out.printf("%-10s %-10s %-10s %-20s\n", "Height(cm)",
"Weight(kg)", "BMI", "Status");

System.out.println("----------------------------------------------------
-");
        for (String[] row : results) {
            System.out.printf("%-10s %-10s %-10s %-20s\n", row[0], row[1],
row[2], row[3]);
        }
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        int numPersons = 10;
        double[][] personData = new double[numPersons][2];
```

```
        System.out.println("Enter weight (kg) and height (cm) for " +
numPersons + " persons:");

        for (int i = 0; i < numPersons; i++) {

            System.out.print("Person " + (i + 1) + " - Weight (kg): ");

            personData[i][0] = scanner.nextDouble();

            System.out.print("Person " + (i + 1) + " - Height (cm): ");

            personData[i][1] = scanner.nextDouble();

        }

        String[][] results = computeBMI(personData);

        displayResults(results);

        scanner.close();

    }

}
```

2. Find unique characters in a string using the charAt() method and display the result

   Hint =>

   a. Create a Method to find the length of the text without using the String method length()

   b. Create a method to Find unique characters in a string using the charAt() method and
      return them as a 1D array. The logic used here is as follows:

      i. Create an array to store the unique characters in the text. The size is the length of
          the text

      ii. Loops to Find the unique characters in the text. Find the unique characters in the text
          using a nested loop. An outer loop iterates through each character and an inner loop
          checks if the character is unique by comparing it with the previous characters. If the
          character is unique, it is stored in the result array

      iii. Create a new array to store the unique characters

   c. Finally, the main function takes user inputs, calls the user-defined methods, and displays
      the result.

```java
package Day5.LabPractice_L3;

import java.util.Scanner;

public class LP2 {

    public static int findStringLength(String text) {

        int count = 0;

        try {

            while (true) {

                text.charAt(count);
```

```java
                    count++;

                }

        } catch (IndexOutOfBoundsException e) {

        }

        return count;

    }

    public static char[] findUniqueCharacters(String text) {

        int length = findStringLength(text);

        char[] uniqueChars = new char[length];

        int uniqueCount = 0;

        for (int i = 0; i < length; i++) {

            char currentChar = text.charAt(i);

            boolean isUnique = true;

            for (int j = 0; j < uniqueCount; j++) {

                if (uniqueChars[j] == currentChar) {

                    isUnique = false;

                    break;

                }

            }

            if (isUnique) {

                uniqueChars[uniqueCount++] = currentChar;

            }

        }

        char[] finalUniqueChars = new char[uniqueCount];

        for (int i = 0; i < uniqueCount; i++) {

            finalUniqueChars[i] = uniqueChars[i];

        }

        return finalUniqueChars;

    }

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter a string: ");

        String inputText = scanner.nextLine();
```

```
        char[] uniqueCharacters = findUniqueCharacters(inputText);

        System.out.println("Unique characters in the string:");

        for (char c : uniqueCharacters) {

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

        }

        scanner.close();

    }

}
```

3. Write a program to find the first non-repeating character in a string and show the result

   Hint =>

   a. Non-repeating character is a character that occurs only once in the string
   b. Create a Method to find the first non-repeating character in a string using the charAt() method and return the character. The logic used here is as follows:
      i. Create an array to store the frequency of characters in the text. ASCII values of characters are used as indexes in the array to store the frequency of each character. There are 256 ASCII characters
      ii. Loop through the text to find the frequency of characters in the text
      iii. Loop through the text to find the first non-repeating character in the text by checking the frequency of each character
   c. In the main function take user inputs, call user-defined methods, and displays result.

```
package Day5.LabPractice_L3;

import java.util.Scanner;

public class LP3 {

    public static char findFirstNonRepeatingChar(String text) {

        int[] charFrequency = new int[256];

        for (int i = 0; i < text.length(); i++) {

            charFrequency[text.charAt(i)]++;

        }

        for (int i = 0; i < text.length(); i++) {

            if (charFrequency[text.charAt(i)] == 1) {

                return text.charAt(i);

            }

        }

        return '\0';
```

```
        }
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a string: ");
        String inputText = scanner.nextLine();
        char result = findFirstNonRepeatingChar(inputText);
        if (result != '\0') {
            System.out.println("First non-repeating character: " +
result);
        } else {
            System.out.println("No non-repeating character found.");
        }
        scanner.close();
    }
}
```

4. Write a program to find the frequency of characters in a string using the charAt() method and display the result

   Hint =>

   a. Create a method to find the frequency of characters in a string using the charAt() method and return the characters and their frequencies in a 2D array. The logic used here is as follows:
      i. Create an array to store the frequency of characters in the text. ASCII values of characters are used as indexes in the array to store the frequency of each character. There are 256 ASCII characters
      ii. Loop through the text to find the frequency of characters in the text
      iii. Create an array to store the characters and their frequencies
      iv. Loop through the characters in the text and store the characters and their frequencies
   b. In the main function take user inputs, call user-defined methods, and displays result.

```
package Day5.LabPractice_L3;
import java.util.Scanner;
public class LP4 {
    public static String[][] findCharacterFrequency(String text) {
        int[] charFrequency = new int[256];
        for (int i = 0; i < text.length(); i++) {
            charFrequency[text.charAt(i)]++;
```

```java
        }
        int uniqueCount = 0;
        for (int i = 0; i < 256; i++) {
            if (charFrequency[i] > 0) {
                uniqueCount++;
            }
        }
        String[][] result = new String[uniqueCount][2];
        int index = 0;
        for (int i = 0; i < 256; i++) {
            if (charFrequency[i] > 0) {
                result[index][0] = Character.toString((char) i);
                result[index][1] = Integer.toString(charFrequency[i]);
                index++;
            }
        }
        return result;
    }
    public static void displayFrequency(String[][] frequencyTable) {
        System.out.println("\nCharacter\tFrequency");
        for (String[] row : frequencyTable) {
            System.out.println("   " + row[0] + "\t\t   " + row[1]);
        }
    }
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a string: ");
        String inputText = scanner.nextLine();
        String[][] frequencyTable = findCharacterFrequency(inputText);
        displayFrequency(frequencyTable);
        scanner.close();
    }
}
```

5. Write a program to find the frequency of characters in a string using unique characters and display the result

Hint =>

a. Create a method to Find unique characters in a string using the charAt() method and return them as a 1D array. Use Nested Loops to find the unique characters in the text

b. Create a method to find the frequency of characters in a string and return the characters and their frequencies in a 2D array. The logic used here is as follows:

  i. Create an array to store the frequency of characters in the text. ASCII values of characters are used as indexes in the array to store the frequency of each character. There are 256 ASCII characters

  ii. Loop through the text to find the frequency of characters in the text

  iii. Call the uniqueCharacters() method to find the unique characters in the text

  iv. Create a 2D String array to store the unique characters and their frequencies.

  v. Loop through the unique characters and store the characters and their frequencies

c. In the main function take user inputs, call user-defined methods, and displays result.

```java
package Day5.LabPractice_L3;

import java.util.Scanner;

public class LP5 {

    public static char[] findUniqueCharacters(String text) {

        int length = text.length();

        char[] uniqueChars = new char[length];

        int uniqueCount = 0;

        for (int i = 0; i < length; i++) {

            char currentChar = text.charAt(i);

            boolean isUnique = true;

            for (int j = 0; j < uniqueCount; j++) {

                if (uniqueChars[j] == currentChar) {

                    isUnique = false;

                    break;

                }

            }

            if (isUnique) {

                uniqueChars[uniqueCount++] = currentChar;

            }

        }

        char[] result = new char[uniqueCount];
```

```java
        System.arraycopy(uniqueChars, 0, result, 0, uniqueCount);

        return result;

    }

    public static String[][] findCharacterFrequency(String text) {

        int[] charFrequency = new int[256];

        for (int i = 0; i < text.length(); i++) {

            charFrequency[text.charAt(i)]++;

        }

        char[] uniqueChars = findUniqueCharacters(text);

        String[][] result = new String[uniqueChars.length][2];

        for (int i = 0; i < uniqueChars.length; i++) {

            result[i][0] = Character.toString(uniqueChars[i]);

            result[i][1] =
Integer.toString(charFrequency[uniqueChars[i]]);

        }


        return result;

    }

    public static void displayFrequency(String[][] frequencyTable) {

        System.out.println("\nCharacter\tFrequency");

        for (String[] row : frequencyTable) {

            System.out.println("   " + row[0] + "\t\t   " + row[1]);

        }

    }

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter a string: ");

        String inputText = scanner.nextLine();

        String[][] frequencyTable = findCharacterFrequency(inputText);

        displayFrequency(frequencyTable);

        scanner.close();

    }

}
```

6. Write a program to find the frequency of characters in a string using nested loops and display the result

Hint =>

a. Create a method to find the frequency of characters in a string and return the characters and their frequencies in a 1D array. The logic used here is as follows:
   i. Create an array to store the frequency of each character in the text and an array to store the characters in the text using the toCharArray() method
   ii. Loops to Find the frequency of each character in the text and store the result in a frequency array. For this use a Nested Loop with an Outer loop to iterate through each character in the text and initialize the frequency of each character to 1. And an Inner loop to check for duplicate characters. In case of duplicate increment the frequency value and set the duplicate characters to '0' to avoid counting them again.
   iii. Create a 1D String array to store the characters and their frequencies. For this Iterate through the characters in the text and store the characters and their frequencies

b. Finally, the main function takes user inputs, calls the user-defined methods, and displays the result.

```java
package Day5.LabPractice_L3;

import java.util.Scanner;

public class Lp6 {

    public static String[] findCharacterFrequency(String text) {

        char[] charArray = text.toCharArray();

        int length = charArray.length;

        int[] frequency = new int[length];

        for (int i = 0; i < length; i++) {

            frequency[i] = 1;

        }

        for (int i = 0; i < length; i++) {

            if (charArray[i] == '0') {

                continue;

            }

            for (int j = i + 1; j < length; j++) {

                if (charArray[i] == charArray[j]) {

                    frequency[i]++;

                    charArray[j] = '0';
```

```
                }

            }

        }

        int count = 0;

        for (char c : charArray) {

            if (c != '0') count++;

        }

        String[] result = new String[count];

        int index = 0;

        for (int i = 0; i < length; i++) {

            if (charArray[i] != '0') {

                result[index++] = charArray[i] + " - " + frequency[i];

            }

        }

        return result;

    }

    public static void displayFrequency(String[] frequencyTable) {

        System.out.println("\nCharacter\tFrequency");

        for (String row : frequencyTable) {

            System.out.println("   " + row.replace(" - ", "\t\t   "));

        }

    }

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter a string: ");

        String inputText = scanner.nextLine();

        String[] frequencyTable = findCharacterFrequency(inputText);

        displayFrequency(frequencyTable);

        scanner.close();

    }

}
```

7. Write a program to to check if a text is palindrome and display the result

Hint =>

a. A palindrome is a word, phrase, number, or other sequence of characters that reads the same forward and backward

b. **Logic 1:** Write a method to compare the characters from the start and end of the string to determine whether the text is palindrome. The logic used here is as follows:
   i. Set the start and end indexes of the text
   ii. Loop through the text and compare the characters from the start and the end of the string. If the characters are not equal, return false

c. **Logic 2:** Write a recursive method to compare the characters from the start and end of the text passed as parameters using recursion. The logic used here is as follows:
   i. First, check if the start index is greater than or equal to the end index, then return true.
   ii. If the characters at the start and end indexes are not equal, return false.
   iii. Otherwise, call the method recursively with the start index incremented by 1 and the end index

d. **Logic 3:** Write a Method to compare the characters from the start and end of the text using character arrays. The logic used here is as follows:
   i. Firstly Write a Method to reverse a string using the charAt() method and return the reversal array.
   ii. Create a character array using the String method toCharArray() and also create a reverse array. Compare the characters in the original and reverse arrays to do a Palindrome check

e. Finally, in the main method do palindrome check using the three logic and display result

```java
package Day5.LabPractice_L3;

import java.util.Scanner;

public class LP7 {

    public static boolean isPalindromeIterative(String text) {

        int start = 0, end = text.length() - 1;

        while (start < end) {

            if (text.charAt(start) != text.charAt(end)) {

                return false;

            }

            start++;

            end--;

        }

        return true;

    }
```

```java
    public static boolean isPalindromeRecursive(String text, int start,
int end) {

        if (start >= end) {

            return true;

        }

        if (text.charAt(start) != text.charAt(end)) {

            return false;

        }

        return isPalindromeRecursive(text, start + 1, end - 1);

    }

    public static boolean isPalindromeUsingCharArray(String text) {

        char[] originalArray = text.toCharArray();

        char[] reversedArray = reverseString(text);

        for (int i = 0; i < originalArray.length; i++) {

            if (originalArray[i] != reversedArray[i]) {

                return false;

            }

        }

        return true;

    }

    public static char[] reverseString(String text) {

        int length = text.length();

        char[] reversed = new char[length];

        for (int i = 0; i < length; i++) {

            reversed[i] = text.charAt(length - i - 1);

        }

        return reversed;

    }

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter a string: ");

        String inputText = scanner.nextLine();

        boolean result1 = isPalindromeIterative(inputText);
```

```java
        boolean result2 = isPalindromeRecursive(inputText, 0,
inputText.length() - 1);

        boolean result3 = isPalindromeUsingCharArray(inputText);

        System.out.println("\nPalindrome Check Results:");

        System.out.println("1. Using Iterative Approach: " + (result1 ?
"Palindrome" : "Not a Palindrome"));

        System.out.println("2. Using Recursion: " + (result2 ?
"Palindrome" : "Not a Palindrome"));

        System.out.println("3. Using Character Array: " + (result3 ?
"Palindrome" : "Not a Palindrome"));

        scanner.close();

    }

}
```

8. Write a program to check if two texts are anagrams and display the result

    Hint =>

    a. An anagram is a word or phrase formed by rearranging the same letters to form different words or phrases,
    b. Write a method to check if two texts are anagrams. The logic used here is as follows:
        i.   Check if the lengths of the two texts are equal
        ii. Create an array to store the frequency of characters in the strings for the two text
        iii. Find the frequency of characters in the two texts using the loop
        iv. Compare the frequency of characters in the two texts. If the frequencies are not equal, return false
    c. In the main function take user inputs, call user-defined methods, and displays result.

```java
package Day5.LabPractice_L3;
import java.util.Scanner;
public class LP8 {
    public static boolean areAnagrams(String text1, String text2) {
        if (text1.length() != text2.length()) {
            return false;
        }
        int[] charFrequency1 = new int[256];
        int[] charFrequency2 = new int[256];
        for (int i = 0; i < text1.length(); i++) {
            charFrequency1[text1.charAt(i)]++;
```

```java
            charFrequency2[text2.charAt(i)]++;
        }

        for (int i = 0; i < 256; i++) {

            if (charFrequency1[i] != charFrequency2[i]) {

                return false;

            }

        }

        return true;

    }

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter first string: ");

        String input1 = scanner.nextLine().toLowerCase();

        System.out.print("Enter second string: ");

        String input2 = scanner.nextLine().toLowerCase();

        boolean result = areAnagrams(input1.replaceAll("\\s", ""),
input2.replaceAll("\\s", "")); // Removing spaces


        if (result) {

            System.out.println("The given texts are anagrams.");

        } else {

            System.out.println("The given texts are NOT anagrams.");

        }

        scanner.close();

    }

}
```

9. Create a program to display a calendar for a given month and year. The program should take the month and year as input from the user and display the calendar for that month. E.g. for 07 2005 user input, the program should display the calendar as shown below

```
July 2005
Sun Mon Tue Wed Thu Fri Sat
                          1   2
  3   4   5   6   7   8   9
 10  11  12  13  14  15  16
 17  18  19  20  21  22  23
 24  25  26  27  28  29  30
 31
```

Hint =>

a. Write a Method to get the name of the month. For this define a month Array to store the names of the months

b. Write a Method to get the number of days in the month. For this define a days Array to store the number of days in each month. For Feb month, check for Leap Year to get the number of days. Also, define a Leap Year Method.

c. Write a method to get the first day of the month using the Gregorian calendar algorithm

$$y0 = y − (14 − m) / 12$$

$$x = y0 + y0/4 − y0/100 + y0/400$$

$$m0 = m + 12 × ((14 − m) / 12) − 2$$

$$d0 = (d + x + 31m0 / 12) \bmod 7$$

d. Displaying the Calendar requires 2 *for* loops.

   i. The first *for* loop up to the first day to get the proper indentation. As in the example above 3 spaces from Sun to Thu as to be set as July 1st starts on Fri

   ii. The Second *for* loop Displays the days of the month starting from 1 to the number of days. Add proper indentation for single-digit days using *%3d* to display the integer right-justified in a field of width 3. Please note to move to the next line after Sat

```java
package Day5.LabPractice_L3;
import java.util.Scanner;
public class LP9 {
    public static String getMonthName(int month) {
        String[] months = {"January", "February", "March", "April", "May", "June",
            "July", "August", "September", "October", "November", "December"};
        return months[month - 1];
    }
    public static boolean isLeapYear(int year) {
        return (year % 4 == 0 && year % 100 != 0) || (year % 400 == 0);
    }
    public static int getDaysInMonth(int month, int year) {
        int[] days = {31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
        if (month == 2 && isLeapYear(year)) {
            return 29;
```

```java
        }
        return days[month - 1];
    }
    public static int getFirstDayOfMonth(int month, int year) {
        int y = year - (14 - month) / 12;
        int x = y + y / 4 - y / 100 + y / 400;
        int m = month + 12 * ((14 - month) / 12) - 2;
        return (1 + x + (31 * m) / 12) % 7;
    }
    public static void displayCalendar(int month, int year) {
        String monthName = getMonthName(month);
        int daysInMonth = getDaysInMonth(month, year);
        int firstDay = getFirstDayOfMonth(month, year);
        System.out.println("\n    " + monthName + " " + year);
        System.out.println(" Sun Mon Tue Wed Thu Fri Sat");
        for (int i = 0; i < firstDay; i++) {
            System.out.print("    ");
        }
        for (int day = 1; day <= daysInMonth; day++) {
            System.out.printf("%4d", day);
            if ((day + firstDay) % 7 == 0) {
                System.out.println();
            }
        }
        System.out.println();
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter month (1-12): ");
        int month = scanner.nextInt();
        System.out.print("Enter year: ");
        int year = scanner.nextInt();
        displayCalendar(month, year);
        scanner.close();
    }
}
```

10. Write a program to create a deck of cards, initialize the deck, shuffle the deck, and distribute the deck of n cards to x number of players. Finally, print the cards the players have.

    Hint =>

    a. Create a deck of cards with suits "Hearts", "Diamonds", "Clubs", "Spades" and ranks from "2", "3", "4", "5", "6", "7", "8", "9", "10", "Jack", "Queen", "King", and "Ace"

    b. Calculate the number of cards in the deck and initialize the deck

    ```java
    int numOfCards = suits.length * ranks.length;
    ```

c. Write a Method to Initialize the deck of cards with suits and ranks and return the deck. The deck is an array of strings where each string represents a card in the deck represented as "rank of suit" e.g., "2 of Hearts"

d. Write a Method to Shuffle the deck of cards and return the shuffled deck. To shuffle the card iterate over the deck and swap each card with a random card from the remaining deck to shuffle the deck. Please find the steps below

Step1: Use for Loop Iterate over the deck and swap each card with a random card from the remaining deck

Step 2: Inside the Loop Generate a random card number between i and n using the following code

```
int randomCardNumber = i + (int) (Math.random() * (n - i));
```

Step 3: Swap the current card with the random card

e. Write a Method to distribute the deck of n cards to x number of players and return the players. For this Check the n cards can be distributed to x players. If possible then Create a 2D array to store the players and their cards

f. Write a Method to Print the players and their cards

```java
package Day5.LabPractice_L3;
import java.util.Random;
import java.util.Scanner;
public class LP10 {
    private static final String[] SUITS = {"Hearts", "Diamonds", "Clubs", "Spades"};
    private static final String[] RANKS = {"2", "3", "4", "5", "6", "7", "8", "9", "10", "Jack", "Queen", "King", "Ace"};
    public static String[] initializeDeck() {
        int numOfCards = SUITS.length * RANKS.length;
        String[] deck = new String[numOfCards];
        int index = 0;
        for (String suit : SUITS) {
            for (String rank : RANKS) {
                deck[index++] = rank + " of " + suit;
            }
        }
        return deck;
    }
```

```java
    public static void shuffleDeck(String[] deck) {

        Random rand = new Random();

        int n = deck.length;

        for (int i = 0; i < n; i++) {

            int randomCardNumber = i + rand.nextInt(n - i);

            String temp = deck[i];

            deck[i] = deck[randomCardNumber];

            deck[randomCardNumber] = temp;

        }

    }

    public static String[][] distributeCards(String[] deck, int
numPlayers, int numCards) {

        int totalCards = numPlayers * numCards;

        if (totalCards > deck.length) {

            System.out.println("Not enough cards to distribute.");

            return null;

        }

        String[][] players = new String[numPlayers][numCards];

        int index = 0;

        for (int i = 0; i < numCards; i++) {

            for (int j = 0; j < numPlayers; j++) {

                players[j][i] = deck[index++];

            }

        }

        return players;

    }

    public static void printPlayersCards(String[][] players) {

        if (players == null) return;

        for (int i = 0; i < players.length; i++) {

            System.out.println("Player " + (i + 1) + "'s cards:");

            for (String card : players[i]) {

                System.out.println("  - " + card);

            }
```

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

        }

    }

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter number of players: ");

        int numPlayers = scanner.nextInt();

        System.out.print("Enter number of cards per player: ");

        int numCards = scanner.nextInt();

        String[] deck = initializeDeck();

        shuffleDeck(deck);

        String[][] players = distributeCards(deck, numPlayers, numCards);

        printPlayersCards(players);

        scanner.close();

    }

}
```