Level 3 String

1. Find BMI of 10 Members

```
import java.util.*;


class BMIProgram {
    public static String[][] calculateBMI(double[][] data) {
        String[][] result = new String[data.length][4];
        for (int i = 0; i < data.length; i++) {
            double weight = data[i][0];
            double height = data[i][1] / 100.0;
            double bmi = weight / (height * height);
            String status;
            if (bmi < 18.5) status = "Underweight";
            else if (bmi < 24.9) status = "Normal";
            else if (bmi < 29.9) status = "Overweight";
            else status = "Obese";
            result[i][0] = String.valueOf(weight);
            result[i][1] = String.valueOf(data[i][1]);
            result[i][2] = String.format("%.2f", bmi);
            result[i][3] = status;
        }
        return result;
    }

    public static void display(String[][] result) {
        System.out.printf("%-10s %-10s %-10s %-15s%n", "Weight", "Height", "BMI", "Status");
        for (String[] row : result) {
            System.out.printf("%-10s %-10s %-10s %-15s%n", row[0], row[1], row[2], row[3]);
        }
```

```
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        double[][] data = new double[10][2];
        for (int i = 0; i < 10; i++) {
            System.out.print("Enter weight (kg) for person " + (i + 1) + ": ");
            data[i][0] = sc.nextDouble();
            System.out.print("Enter height (cm) for person " + (i + 1) + ": ");
            data[i][1] = sc.nextDouble();
        }
        String[][] result = calculateBMI(data);
        display(result);
    }
}
```

2. Find Unique Characters in a String

```
import java.util.*;

class UniqueCharacters {
    public static int getLength(String text) {
        int count = 0;
        try {
            while (true) {
                text.charAt(count);
                count++;
            }
        } catch (Exception e) {}
        return count;
```

```
    }

    public static char[] findUnique(String text) {
        int n = getLength(text);
        char[] unique = new char[n];
        int index = 0;
        for (int i = 0; i < n; i++) {
            char c = text.charAt(i);
            boolean isUnique = true;
            for (int j = 0; j < i; j++) {
                if (c == text.charAt(j)) {
                    isUnique = false;
                    break;
                }
            }
            if (isUnique) unique[index++] = c;
        }
        return Arrays.copyOf(unique, index);
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        String text = sc.nextLine();
        char[] unique = findUnique(text);
        System.out.println("Unique characters: " + Arrays.toString(unique));
    }
}
```

3. First Non-Repeating Character

```
import java.util.*;

class FirstNonRepeating {
    public static char findFirstNonRepeating(String text) {
        int[] freq = new int[256];
        for (int i = 0; i < text.length(); i++) {
            freq[text.charAt(i)]++;
        }
        for (int i = 0; i < text.length(); i++) {
            if (freq[text.charAt(i)] == 1) return text.charAt(i);
        }
        return 0;
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        String text = sc.nextLine();
        char result = findFirstNonRepeating(text);
        if (result != 0) System.out.println("First non-repeating: " + result);
        else System.out.println("No non-repeating character found.");
    }
}
```

4. Frequency of Characters using charAt

```
import java.util.*;

class FrequencyCharAt {
    public static String[][] frequency(String text) {
        int[] freq = new int[256];
```

```
        for (int i = 0; i < text.length(); i++) {
            freq[text.charAt(i)]++;
        }
        String[][] result = new String[text.length()][2];
        int index = 0;
        for (int i = 0; i < text.length(); i++) {
            char c = text.charAt(i);
            if (freq[c] != 0) {
                result[index][0] = String.valueOf(c);
                result[index][1] = String.valueOf(freq[c]);
                freq[c] = 0;
                index++;
            }
        }
        return Arrays.copyOf(result, index);
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        String text = sc.nextLine();
        String[][] result = frequency(text);
        for (String[] row : result) {
            System.out.println(row[0] + " -> " + row[1]);
        }
    }
}
```

5. Frequency of Characters using Unique Characters

```
import java.util.*;
```

```
class FrequencyUnique {
    public static char[] uniqueChars(String text) {
        int n = text.length();
        char[] arr = new char[n];
        int index = 0;
        for (int i = 0; i < n; i++) {
            char c = text.charAt(i);
            boolean seen = false;
            for (int j = 0; j < i; j++) {
                if (text.charAt(j) == c) {
                    seen = true;
                    break;
                }
            }
            if (!seen) arr[index++] = c;
        }
        return Arrays.copyOf(arr, index);
    }

    public static String[][] frequency(String text) {
        int[] freq = new int[256];
        for (int i = 0; i < text.length(); i++) freq[text.charAt(i)]++;
        char[] unique = uniqueChars(text);
        String[][] result = new String[unique.length][2];
        for (int i = 0; i < unique.length; i++) {
            result[i][0] = String.valueOf(unique[i]);
            result[i][1] = String.valueOf(freq[unique[i]]);
        }
```

```
        return result;
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        String text = sc.nextLine();
        String[][] result = frequency(text);
        for (String[] row : result) {
            System.out.println(row[0] + " -> " + row[1]);
        }
    }
}
```

6. Frequency of Characters using Nested Loops

```
import java.util.*;

class FrequencyNested {
    public static String[] frequency(String text) {
        char[] chars = text.toCharArray();
        int[] freq = new int[chars.length];
        for (int i = 0; i < chars.length; i++) {
            if (chars[i] == '0') continue;
            freq[i] = 1;
            for (int j = i + 1; j < chars.length; j++) {
                if (chars[i] == chars[j]) {
                    freq[i]++;
                    chars[j] = '0';
                }
            }
```

```
        }
        String[] result = new String[chars.length];
        int index = 0;
        for (int i = 0; i < chars.length; i++) {
            if (chars[i] != '0') {
                result[index++] = chars[i] + " -> " + freq[i];
            }
        }
        return Arrays.copyOf(result, index);
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        String text = sc.nextLine();
        String[] result = frequency(text);
        for (String s : result) System.out.println(s);
    }
}
```

7. Palindrome Check

```
import java.util.*;

class Palindrome {
    public static boolean isPalindrome1(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;
    }

    public static boolean isPalindrome2(String text, int start, int end) {
        if (start >= end) return true;
        if (text.charAt(start) != text.charAt(end)) return false;
        return isPalindrome2(text, start + 1, end - 1);
    }

    public static boolean isPalindrome3(String text) {
        char[] arr = text.toCharArray();
        char[] rev = new char[arr.length];
        for (int i = 0; i < arr.length; i++) rev[i] = text.charAt(arr.length - 1 - i);
        return Arrays.equals(arr, rev);
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        String text = sc.nextLine();
        System.out.println("Method1: " + isPalindrome1(text));
        System.out.println("Method2: " + isPalindrome2(text, 0, text.length() - 1));
        System.out.println("Method3: " + isPalindrome3(text));
    }
}
```

8. Anagram Check

```
import java.util.*;
```

```
class Anagram {
    public static boolean isAnagram(String s1, String s2) {
        if (s1.length() != s2.length()) return false;
        int[] freq1 = new int[256];
        int[] freq2 = new int[256];
        for (char c : s1.toCharArray()) freq1[c]++;
        for (char c : s2.toCharArray()) freq2[c]++;
        return Arrays.equals(freq1, freq2);
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        String s1 = sc.nextLine();
        String s2 = sc.nextLine();
        System.out.println("Anagram? " + isAnagram(s1, s2));
    }
}
```

9. Calendar Program

```
import java.util.*;

class CalendarProgram {
    public static String getMonthName(int m) {
        String[] months = {"January","February","March","April","May","June","July",
                           "August","September","October","November","December"};
        return months[m - 1];
    }

    public static int getDays(int m, int y) {
```

```
    int[] days = {31,28,31,30,31,30,31,31,30,31,30,31};
    if (m == 2 && isLeapYear(y)) return 29;
    return days[m - 1];
}

public static boolean isLeapYear(int y) {
    return (y % 4 == 0 && y % 100 != 0) || (y % 400 == 0);
}

public static int getFirstDay(int d, int m, int y) {
    int y0 = y - (14 - m) / 12;
    int x = y0 + y0/4 - y0/100 + y0/400;
    int m0 = m + 12 * ((14 - m) / 12) - 2;
    return (d + x + 31*m0/12) % 7;
}

public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    int m = sc.nextInt();
    int y = sc.nextInt();
    int days = getDays(m, y);
    int start = getFirstDay(1, m, y);
    System.out.println(getMonthName(m) + " " + y);
    System.out.println("Sun Mon Tue Wed Thu Fri Sat");
    for (int i = 0; i < start; i++) System.out.print("    ");
    for (int d = 1; d <= days; d++) {
        System.out.printf("%3d ", d);
        if (((d + start) % 7 == 0) || d == days) System.out.println();
    }
```

```
    }
}
```

10. Deck of Cards

```
import java.util.*;

class DeckOfCards {
    public static String[] initializeDeck() {
        String[] suits = {"Hearts","Diamonds","Clubs","Spades"};
        String[] ranks = {"2","3","4","5","6","7","8","9","10","Jack","Queen","King","Ace"};
        String[] deck = new String[suits.length * ranks.length];
        int index = 0;
        for (String s : suits) {
            for (String r : ranks) {
                deck[index++] = r + " of " + s;
            }
        }
        return deck;
    }

    public static void shuffle(String[] deck) {
        for (int i = 0; i < deck.length; i++) {
            int rand = i + (int)(Math.random() * (deck.length - i));
            String temp = deck[i];
            deck[i] = deck[rand];
            deck[rand] = temp;
        }
    }
```

```
    public static String[][] distribute(String[] deck, int n, int players) {
        if (n % players != 0) return null;
        String[][] result = new String[players][n / players];
        int index = 0;
        for (int i = 0; i < players; i++) {
            for (int j = 0; j < n / players; j++) {
                result[i][j] = deck[index++];
            }
        }
        return result;
    }

    public static void print(String[][] players) {
        for (int i = 0; i < players.length; i++) {
            System.out.println("Player " + (i+1) + ": " + Arrays.toString(players[i]));
        }
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        String[] deck = initializeDeck();
        shuffle(deck);
        int n = sc.nextInt();
        int players = sc.nextInt();
        String[][] distributed = distribute(deck, n, players);
        if (distributed != null) print(distributed);
        else System.out.println("Cannot distribute cards evenly.");
    }
}
```