LAB REPORT

VIVA

REPRESENT BY:

NEWBIE MEMERS

No	Name	Matric Number
1	TITUS TAN YU FAN	23004883
2	WONG HOONG LIANG	23053016
3	SIM JING JIA	23004919
4	LIM HONG YU	23004973
5	SOON MING HONG	23004950

Question 1:

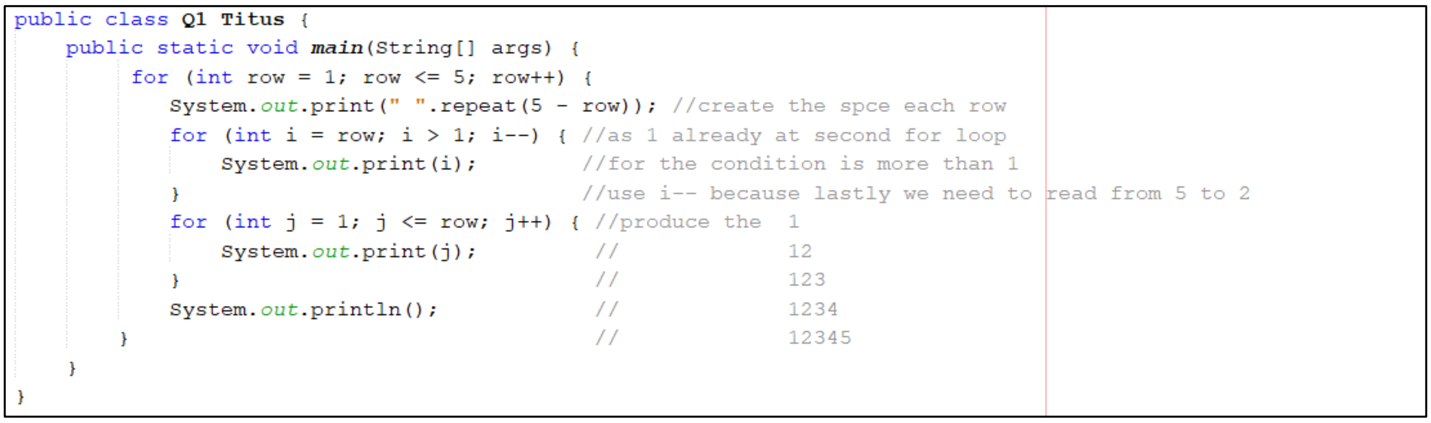
1. Write a program to print the following pattern:

```
    1
   212
  32123
 4321234
543212345
```

Problem description:

To print out 5 row of number pyramid.

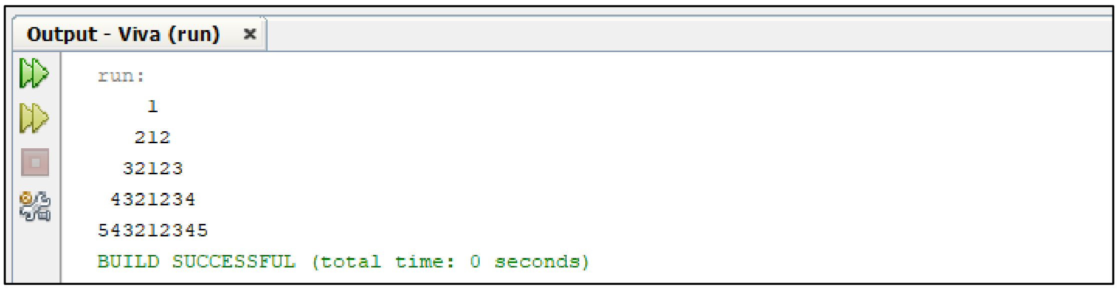
Solution:



```
public class Q1 Titus {
    public static void main(String[] args) {
        for (int row = 1; row <= 5; row++) {
            System.out.print(" ".repeat(5 - row)); //create the spce each row
            for (int i = row; i > 1; i--) { //as 1 already at second for loop
                System.out.print(i);        //for the condition is more than 1
            }                               //use i-- because lastly we need to read from 5 to 2
            for (int j = 1; j <= row; j++) { //produce the  1
                System.out.print(j);         //             12
            }                                //             123
            System.out.println();            //             1234
        }                                    //             12345
    }
}
```

To generate the pyramid following sequences, we used for loop 3 times. The first for loop limit the number of rows will be printed, that's 5. And the first loop creates the space for each row before the numbers printed. For example, the first row will print 4 spaces before print the number. The purpose of this for loop is to make the pyramid shape. After the first loop runs the first time, the program will proceed to second for loop. The second for loop is to print the left-hand side of the pyramid. This loop will not be executed for the first loop as the number 1 already at second for loop, so the condition is more than 1. The update expression uses i-- because lastly we need to read from 5 to 2. After that, third loop will be executed to print the right-hand side and the number 1 of the pyramid. After third loop, it will print a new line and go again the first for loop.

Sample Output:



```
run:
    1
   212
  32123
 4321234
543212345
BUILD SUCCESSFUL (total time: 0 seconds)
```

Question 2:

2. Write a program to reverse the digits of a positive or negative integer.

Sample Output
Enter an integer: -12345 Reversed integer: -54321

Problem description:

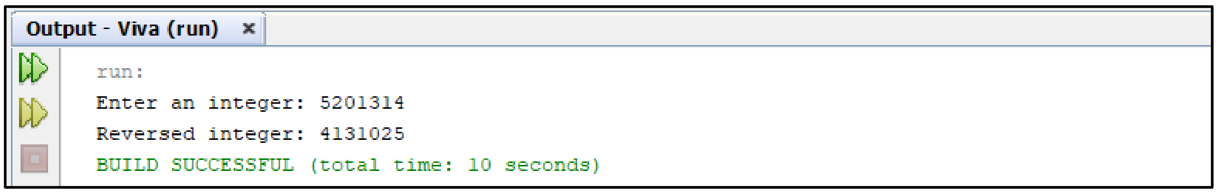
Reverse the number that input by user neither positive nor negative.

Solution:

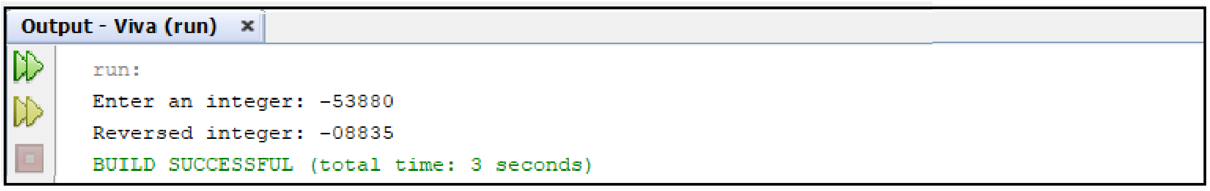
```
import java.util.Scanner;
public class Q2_HoongLiang {

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter an integer: ");
        int in = sc.nextInt();
        System.out.print("Reversed integer: ");

        if (in > 0) {
            for (int IN = in; IN > 0; IN /= 10) {
                System.out.print(IN % 10);
            }
        } else if (in == 0) {
            System.out.print(0);
        } else {
            System.out.print("-");
            for (int IN = in * -1; IN > 0; IN /= 10) {
                System.out.print(IN % 10);
            }
        }
        System.out.println();
    }
}
```

We input a scanner to read what user input. If the input > 0, it will run a for loop. If the input is 0, it will print 0. If the input < 0, it will print negative sign " - " and run the loop same with the first for loop. In the for loop, the input (IN) will undergo modulus division and after that divide by 10 until the IN do not achieve the condition (IN > 0). In the first loop, the IN will undergo modulus division and the result is printed. After that it will run again the loop by divide IN by 10. For example, input 123 undergo first loop and printed 3 and run again, the input 123 divide 10 left 12 and undergo modulus division and get 2. This for loop will run 3 times to get output 321 in this example. Lastly, the system.out.println() is to print a new line for "BUILD SUCCESSFUL". //The maximum and minimum integer is -2147483648 and 2147483648 so the input that excess these will cause error in the program.

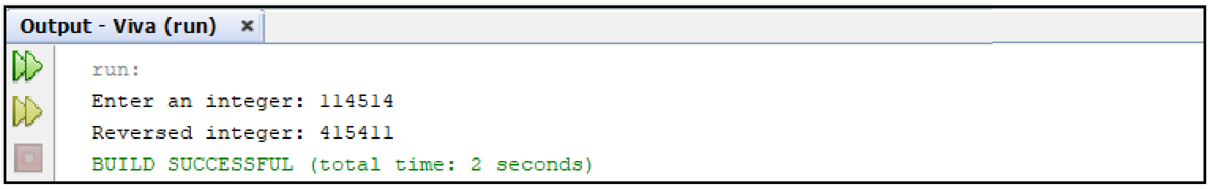
Sample input and output:



```
run:
Enter an integer: 5201314
Reversed integer: 4131025
BUILD SUCCESSFUL (total time: 10 seconds)
```



```
run:
Enter an integer: -53880
Reversed integer: -08835
BUILD SUCCESSFUL (total time: 3 seconds)
```



```
run:
Enter an integer: 114514
Reversed integer: 415411
BUILD SUCCESSFUL (total time: 2 seconds)
```

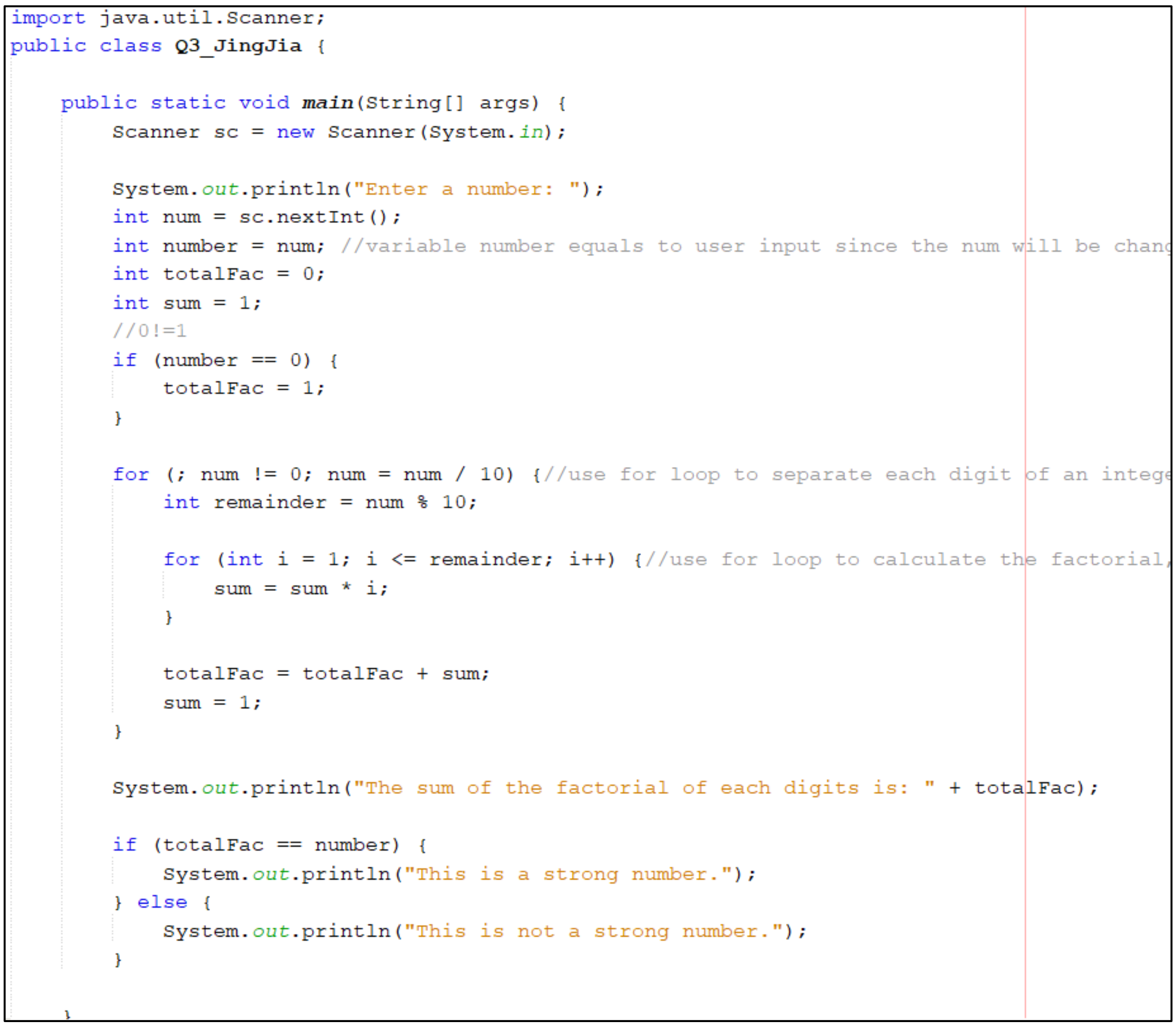
Question 3:

3. Write a program to determine if the number entered by the user is a Strong Number or not. A Strong Number is a number whose sum of the factorial of each digit is equal to the number itself. For example, 145 is a Strong Number because 1! + 4! + 5! = 1 + 24 + 120 = 145.

Problem description:

Determine the number is Strong Number or not.

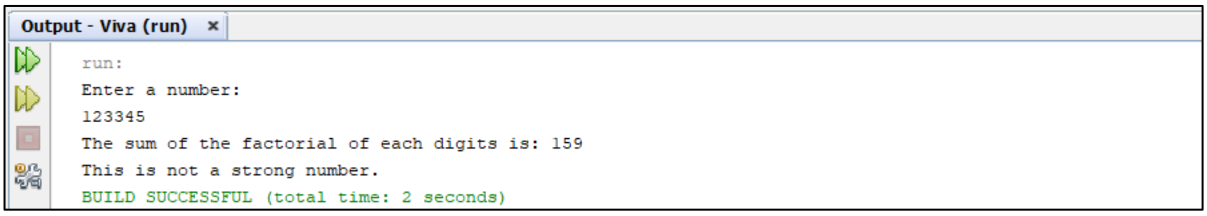
Solution:



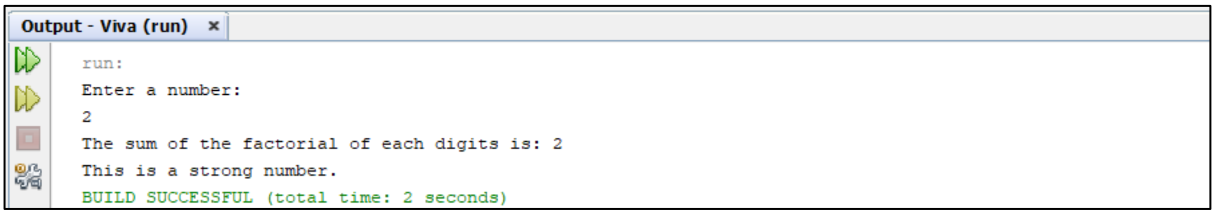
```java
import java.util.Scanner;
public class Q3_JingJia {

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

        System.out.println("Enter a number: ");
        int num = sc.nextInt();
        int number = num; //variable number equals to user input since the num will be chang
        int totalFac = 0;
        int sum = 1;
        //0!=1
        if (number == 0) {
            totalFac = 1;
        }

        for (; num != 0; num = num / 10) {//use for loop to separate each digit of an intege
            int remainder = num % 10;

            for (int i = 1; i <= remainder; i++) {//use for loop to calculate the factorial,
                sum = sum * i;
            }

            totalFac = totalFac + sum;
            sum = 1;
        }

        System.out.println("The sum of the factorial of each digits is: " + totalFac);

        if (totalFac == number) {
            System.out.println("This is a strong number.");
        } else {
            System.out.println("This is not a strong number.");
        }

    }
```

We input a scanner to read the number entered by user. At first, we declare number = num because the num will be changing through the program. We declare sum=1 because if it is 0 then the multiplication will be 0 in the second for loop. We use two for loop to count the factorial of each digit in the number. The first for loop is to separate each digits of the integer by divide the number by 10 and get the remainder. The loop repeat until the number become 0. The second for loop is to calculate the factorial of the digit. It starts with i=1 and repeat until 1 less than or equal to the remainder. The factorial will be sum up after the second for loop. And the sum=1 below it is to reset because the sum become another number during the calculation. If without reset the sum=1, the calculation for the next factorial of digit will incorrect. For example, an input 1234 is entered by user. In the first for loop, 1234 will undergo modulus division and get 4 for the first time. The digit 4 then go to the second for loop to calculate its factorial. The for loop will run 4 times. In the first time, 1=1x1. Second time, 2=1x2. Third time, 6=2x3. Fourth time, 24=6x4. The second for loop will stop and the sum 24 is added into totalFac. The sum 24 also reset to 1 after this to start next round of first loop. That's how we calculate the each digits of the number. We put a condition that if input 0 is entered, the totalFac will output 1. This is because the two for loop can't calculate when input is 0. After that, the calculation result will be print out follow by the determination of the number is strong or not. If the total sum of the factorial of each digit of the number is same to the input, then it will show "This is a strong number." Vice versa.

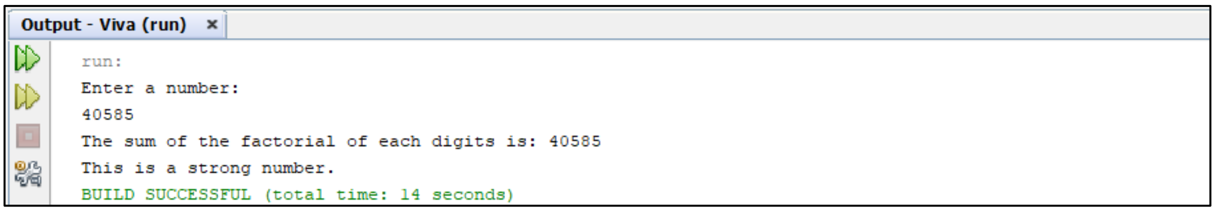
Sample input and output:



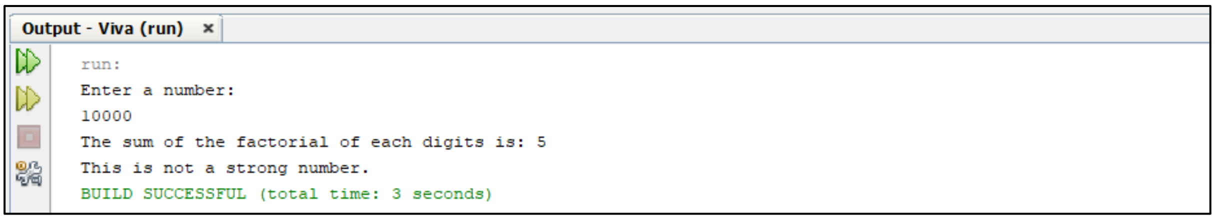
```
run:
Enter a number:
123345
The sum of the factorial of each digits is: 159
This is not a strong number.
BUILD SUCCESSFUL (total time: 2 seconds)
```



```
run:
Enter a number:
2
The sum of the factorial of each digits is: 2
This is a strong number.
BUILD SUCCESSFUL (total time: 2 seconds)
```



```
run:
Enter a number:
40585
The sum of the factorial of each digits is: 40585
This is a strong number.
BUILD SUCCESSFUL (total time: 14 seconds)
```



```
run:
Enter a number:
10000
The sum of the factorial of each digits is: 5
This is not a strong number.
BUILD SUCCESSFUL (total time: 3 seconds)
```

Question 4:

4. A password is said to be strong if it satisfies the following criteria:

   It contains at least one lowercase English character.
   It contains at least one uppercase English character.
   It contains at least one special character. The special characters are: !@#$%^&*()-+
   Its length is at least 8.
   It contains at least one digit.

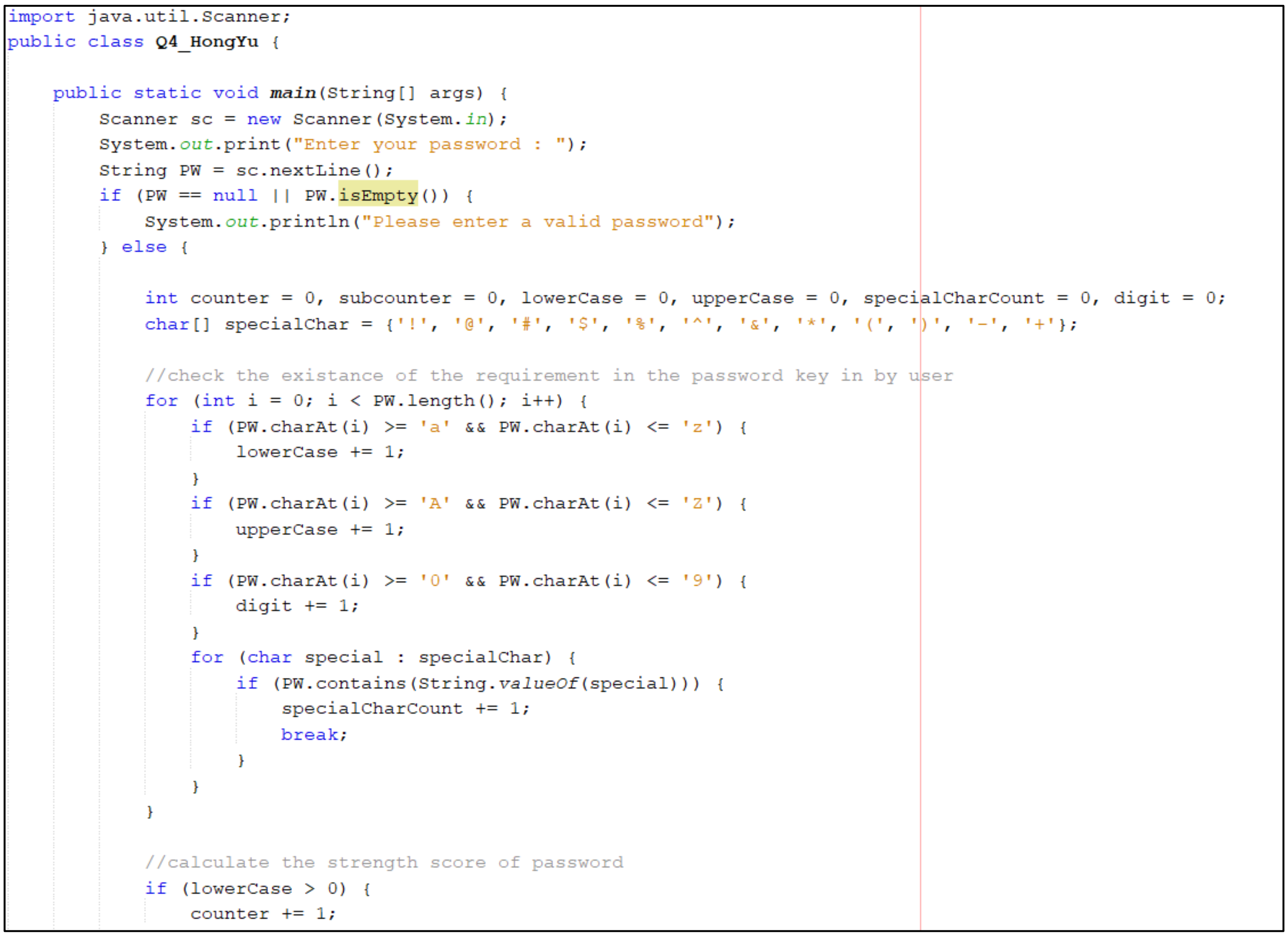
   Check the strength of the input string. Let a strong password be one that satisfies all the above conditions. A moderate password is one that satisfies the first three conditions and has a length of at least 6. Otherwise, the password is weak.

Sample Output
Enter a string: A@b1Cdef Strength of password: Strong

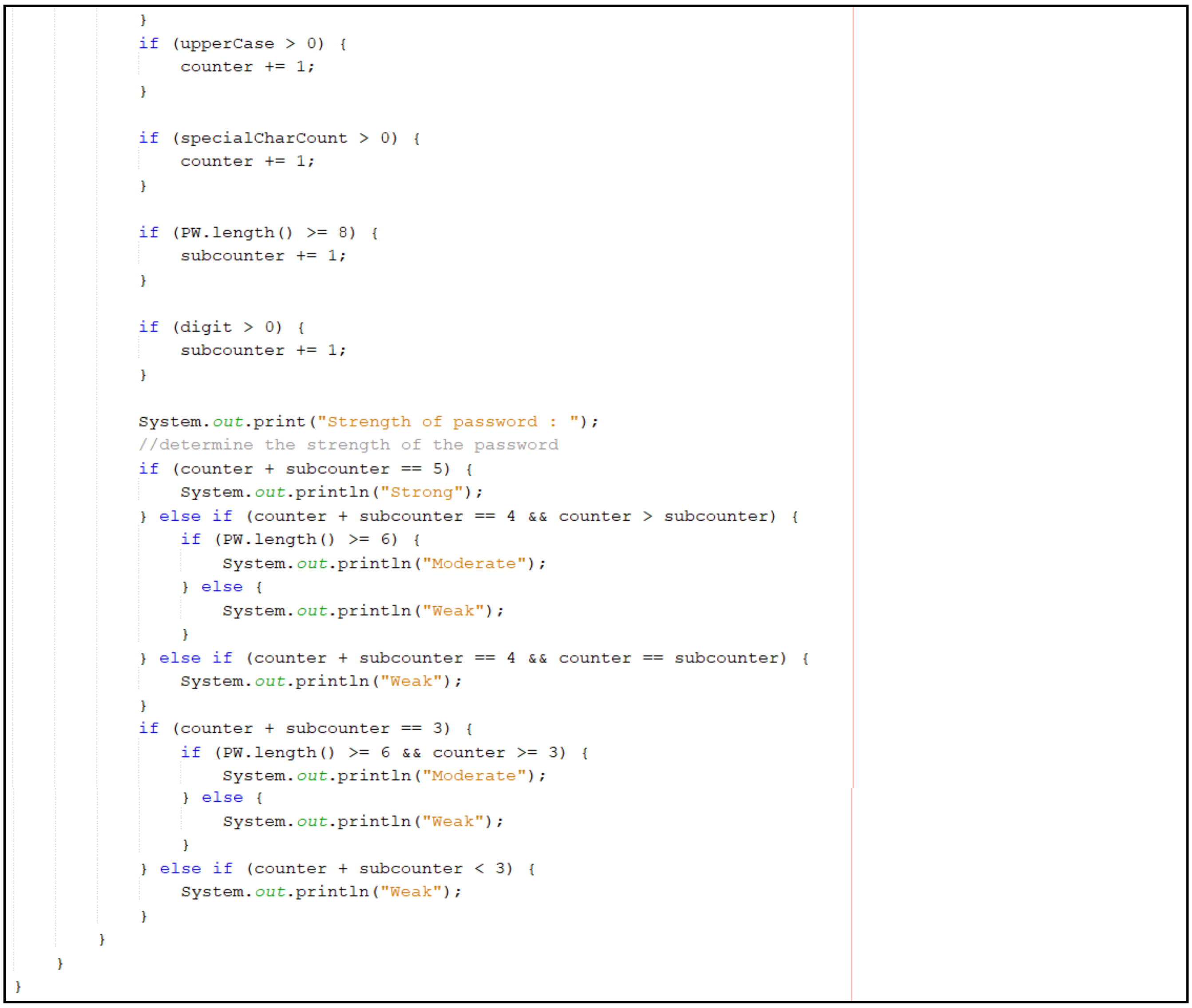
Problem description:

Determine the password is strong, moderate or weak by 5 conditions that stated.

Solution:



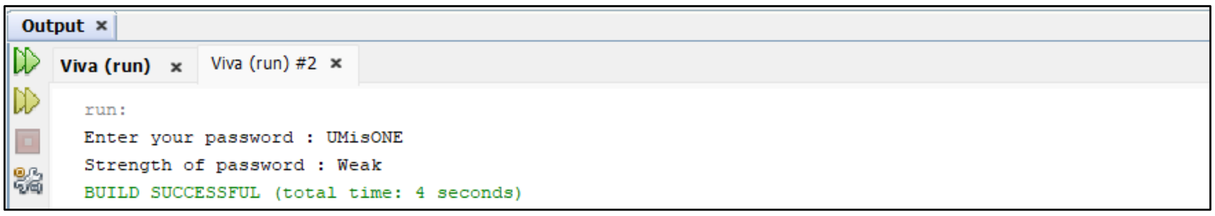
```java
import java.util.Scanner;
public class Q4_HongYu {

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter your password : ");
        String PW = sc.nextLine();
        if (PW == null || PW.isEmpty()) {
            System.out.println("Please enter a valid password");
        } else {

            int counter = 0, subcounter = 0, lowerCase = 0, upperCase = 0, specialCharCount = 0, digit = 0;
            char[] specialChar = {'!', '@', '#', '$', '%', '^', '&', '*', '(', ')', '-', '+'};

            //check the existance of the requirement in the password key in by user
            for (int i = 0; i < PW.length(); i++) {
                if (PW.charAt(i) >= 'a' && PW.charAt(i) <= 'z') {
                    lowerCase += 1;
                }
                if (PW.charAt(i) >= 'A' && PW.charAt(i) <= 'Z') {
                    upperCase += 1;
                }
                if (PW.charAt(i) >= '0' && PW.charAt(i) <= '9') {
                    digit += 1;
                }
                for (char special : specialChar) {
                    if (PW.contains(String.valueOf(special))) {
                        specialCharCount += 1;
                        break;
                    }
                }
            }

            //calculate the strength score of password
            if (lowerCase > 0) {
                counter += 1;
```



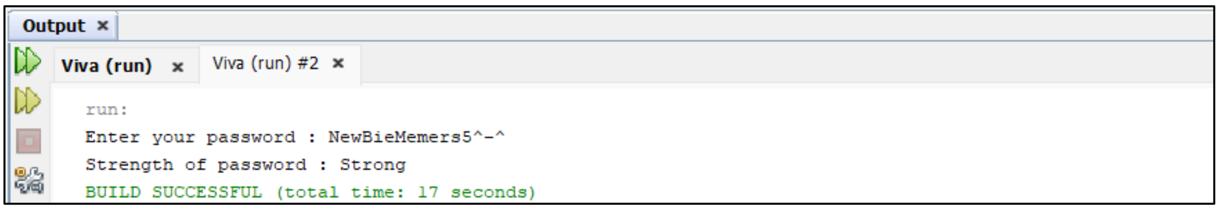
```java
            }
            if (upperCase > 0) {
                counter += 1;
            }

            if (specialCharCount > 0) {
                counter += 1;
            }

            if (PW.length() >= 8) {
                subcounter += 1;
            }

            if (digit > 0) {
                subcounter += 1;
            }

            System.out.print("Strength of password : ");
            //determine the strength of the password
            if (counter + subcounter == 5) {
                System.out.println("Strong");
            } else if (counter + subcounter == 4 && counter > subcounter) {
                if (PW.length() >= 6) {
                    System.out.println("Moderate");
                } else {
                    System.out.println("Weak");
                }
            } else if (counter + subcounter == 4 && counter == subcounter) {
                System.out.println("Weak");
            }
            if (counter + subcounter == 3) {
                if (PW.length() >= 6 && counter >= 3) {
                    System.out.println("Moderate");
                } else {
                    System.out.println("Weak");
                }
            } else if (counter + subcounter < 3) {
                System.out.println("Weak");
            }
        }
    }
}
```

To read the input, we put a scanner. If the input is empty or nothing, it will end the program by showing “Please enter a valid password”, else the input goes into the first for loop. We define variable and declare an array for the special character. The input goes into first loop and will run based on the length of the input. In the first loop, the variable that declared will plus according to the 3 if statement and 1 for loop. For example, the input has 3 “a-z”, the variable “lowerCase” will plus 3. The second loop is to count the number of special characters by using array. It will break the loop if the condition is achieved. After the first loop is run finish based on the length of the input, it’s time to determine the strength of the input, or password. Before determining, we calculate the strength of the password first. Due to the strong and moderate password have the same 3 condition, so we let it be variable “counter” and the rest is “subcounter”. The counter and subcounter is calculated based on the if statement. The condition of if statement is the variable of the first for loop. After calculating the strength of the password, we determine it by using if statement! If the 5 conditions are achieved (3 counter and 2 subcounter), then the password is strong. If the password is achieve 3 counter and 1 subcounter (4 conditions) or 3 counter and 0 subcounter (3 conditions) and its length is equal or greater than 6, than it's moderate. The rest is weak password. That's all.

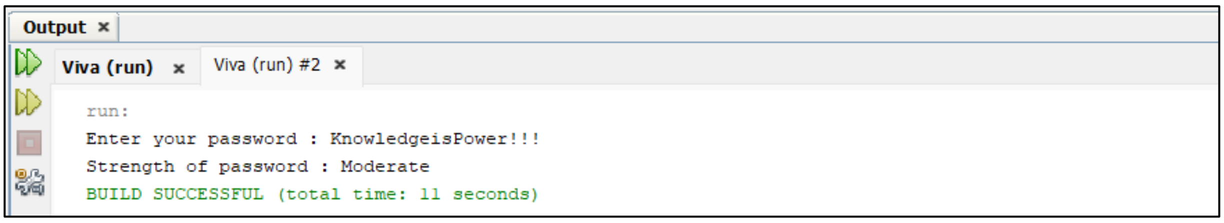
Sample input and output:



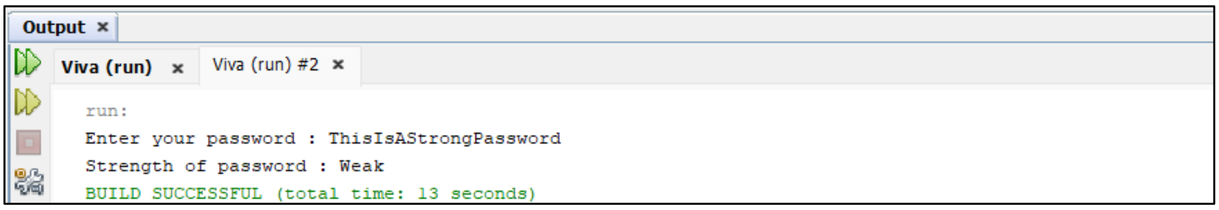
```
run:
Enter your password : UMisONE
Strength of password : Weak
BUILD SUCCESSFUL (total time: 4 seconds)
```



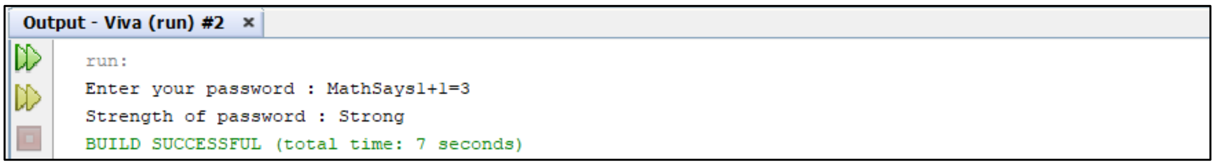
```
run:
Enter your password : NewBieMemers5^-^
Strength of password : Strong
BUILD SUCCESSFUL (total time: 17 seconds)
```



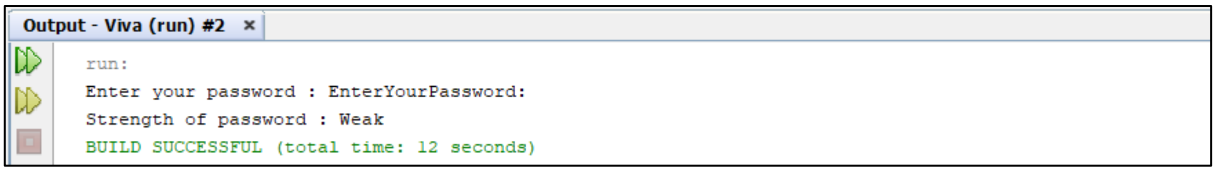
```
run:
Enter your password : KnowledgeisPower!!!
Strength of password : Moderate
BUILD SUCCESSFUL (total time: 11 seconds)
```



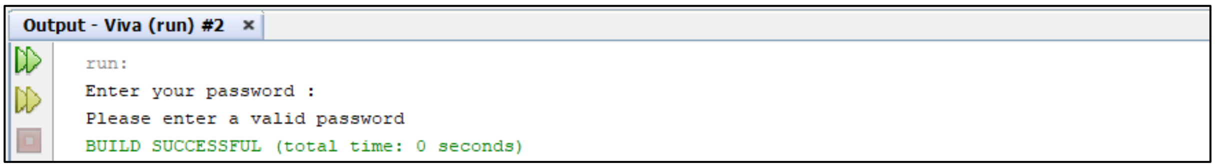
```
run:
Enter your password : ThisIsAStrongPassword
Strength of password : Weak
BUILD SUCCESSFUL (total time: 13 seconds)
```



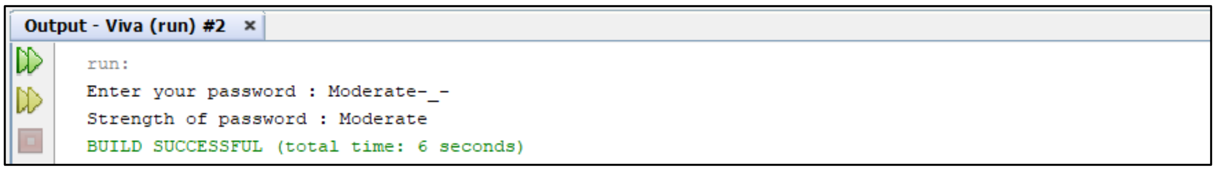
```
run:
Enter your password : MathSays1+1=3
Strength of password : Strong
BUILD SUCCESSFUL (total time: 7 seconds)
```



```
run:
Enter your password : EnterYourPassword:
Strength of password : Weak
BUILD SUCCESSFUL (total time: 12 seconds)
```



```
run:
Enter your password :
Please enter a valid password
BUILD SUCCESSFUL (total time: 0 seconds)
```



```
run:
Enter your password : Moderate-_-
Strength of password : Moderate
BUILD SUCCESSFUL (total time: 6 seconds)
```