Q1. **Problem statement:**

Write a java program to find whether the password is valid or invalid using the regular expression.

Note:

1. Password should be less than or equal to 15 and more than 8 characters in length.
2. Password should contain at least one upper case and one lower case alphabet.
3. Password should contain at least one number.
4. Password should contain at least one special character.

Input Format

The input consists of a string that is in password form.

Output Format

The output displays whether the password is valid or invalid.

Sample Input

```
Iamneo@1
```

Sample Output

```
Iamneo@1 is a valid password
```

Sample Input

```
Iamneo
```

Sample Output

```
Iamneo is a invalid password
```

Time Limit: - ms Memory Limit: - kb Code Size: - kb

Q2. **Problem statement:**

Write a java program to check whether the given content is present in the pattern or not using the regex concept.

Input Format

The input consists of 2 strings. The first one is content and the second one is a pattern.

Output Format

The output prints the true or false. And with the content and pattern.

Sample Input

```
Iamneo test
.*test*.
```

Sample Output

```
Iamneo test contains .*test*. : true
```

Sample Input

```
I am a JAVA Programmer
.*the*.
```

Sample Output

```
I am a JAVA Programmer contains .*the*. : false
```

Time Limit: - ms Memory Limit: - kb Code Size: - kb

Q3. **Problem statement :**
Write a Java program to check whether a string contains only a certain set of characters (in this case a-z, A-Z, and 0-9)

Input Format

The input consists of strings.

Output Format

The output prints the input and returns the match whether it is True or False. Refer to the sample output for the formatting specifications.

Sample Input

```
ABCDEFabcdef123456
```

Sample Output

```
ABCDEFabcdef123456 true
```

Sample Input

```
W3.com
```

Sample Output

```
W3.com false
```

Time Limit: - ms Memory Limit: - kb Code Size: - kb

Q4. **Problem statement:**
Write a Java program to check for a number at the end of a given string.

Input Format

The input consists of the string.

Output Format

The output prints the input text and results with whether the match is found or not found. Refer to the sample output for the formatting specifications.

Sample Input

```
abcdef
```

Sample Output

```
abcdef
Not matched!
```

Sample Input

```
abcdef3459
```

Sample Output

```
abcdef3459 Found a match!
```

Time Limit: - ms Memory Limit: - kb Code Size: - kb

Q5. **Problem statement:**
Write a Java program to count the number of vowels in a given string using regular expressions.

Input Format

The input consists of the string.

Output Format

The output consists of the original string and a new string with the count of the number of vowels. Refer to the sample output for the formatting specifications.

Sample Input

```
Java
```

Sample Output

```
Original string: Java New string: 2
```

Sample Input

```
MID-CENTRALIZED
```

Sample Output

```
Original string: MID-CENTRALIZED New string: 5
```

Time Limit: - ms Memory Limit: - kb Code Size: - kb

Q6. **Problem statement:**
Write a java program to find the number of occurrences of characters from the two strings.

Input Format

The input consists of two strings.

Output Format

The output prints the count of the number of occurrences. Refer to the sample input and output for the formatting specifications.

Sample Input

```
ab abbbabbaba
```

Sample Output

```
The no of occurences: 3
```

Sample Input

```
a aaabbbaaa
```

Sample Output

```
The no of occurences: 6
```

Time Limit: - ms Memory Limit: - kb Code Size: - kb

Q7. **Problem statement:**
Write a regular expression to represent all valid identifiers in java language.
Rules:
The allowed characters are:
1. a to z, A to Z, 0 to 9, -,#
2. The 1st character should be an alphabet symbol only.
3. The length of the identifier should be at least 2.

Sample Input

```
ashok
```

Sample Output

```
ashok:Valid Identifier
```

Sample Input

```
?ashok
```

Sample Output

```
?ashok:Invalid Identifier
```

Time Limit: - ms Memory Limit: - kb Code Size: - kb

Q8. **Problem statement:**
Write a regular expression to represent all mobile numbers.
1. Should contain exactly 10 digits.
2. The 1st digit should be 7 to 9.

Input Format

The input consists of digits.

Output Format

The output prints the mobile number by checking is it a valid number or an invalid number. Refer to the sample input and output for formatting specifications.

Sample Input

```
9989123456
```

Sample Output

```
9989123456 : Valid Number
```

Sample Input

```
6989654321
```

Sample Output

```
6989654321 : Invalid Number
```

Time Limit: - ms Memory Limit: - kb Code Size: - kb

Q9. **Problem statement:**
Write a java program finding data type of user input using Regular Expression.

Input Format

The input consists of different inputs of integer, string, double and date with different formats. Date formats :

1. dd/mm/yyyy : eg: 01/12/2022
2. mm/dd/yyyy : eg: 12/24/2022
3. dd-mon-yy : eg: 01-apr-22
4. dd-mon-yyyy : eg: 01-apr-2022
5. dd-month-yy: eg:01-april-22
6. dd-month-yyyy: eg: 01-april-2022

Output Format

The output prints the value with the appropriate datatype. Refer to the sample input and output for the formatting specifications.

Sample Input

```
100
```

Sample Output

```
The datatype of 100 is: java.lang.Integer
```

Sample Input

```
52.87
```

Sample Output

```
The datatype of 52.87 is: java.lang.Double
```

Sample Input

```
21-apr-1994
```

Sample Output

```
The datatype of 21-apr-1994 is: java.util.Date
```

Sample Input

```
Born to win
```

Sample Output

```
The datatype of Born to win is: java.lang.String
```

Time Limit: - ms Memory Limit: - kb Code Size: - kb

Q10. **Problem statement:**
Write a Java Program to Extract a Single Quote Enclosed String From a Larger String using Regex.

Input Format

The input consists of two strings with single quotes.

Output Format

The output prints the string of the extracted string from the single quote. Refer to the sample input and output for the formatting specifications.

Sample Input

```
Finish what you 'start' I will 'Finish'
```

Sample Output

```
First Extracted part: start Second Extracted part: Finish
```

Sample Input

```
Action speaks louder than 'words' Action 'speaks'
```

Sample Output

```
First Extracted part: words Second Extracted part: speaks
```

Time Limit: - ms Memory Limit: - kb Code Size: - kb