**Re → regular expression**

* **based on given search paatern content is searched**

**Symbols used to generate search pattern**

**\* → match the proceeding char for 0 or more times(infinte)**

**ab\*c**

**ac**

**abc**

**abbc**

**abbbbbbbbbc**

**+ →match the proceeding char for 1 or more times(infinte)**

**ab+c**

**abc**

**abbc**

**abbbc**

**abbbbbc**

**{} → repeat the char or set of chars as many time as the value specified in the brackets**

**ab{3}c --> abbbc**

**ab{5,} → abbbbbc,abbbbbbbbbc**

**ab{3,7} → abbbc,abbbbc,abbbbbc,abbbbbbc,abbbbbbbc**

**. → any symbol**

**a.c → abc,a2c,a!c**

**a.\*C → abbbbc**

**? → proceeding char may or may not present in the string**

**demo\.docx?-→ collectively**

**^ → starting position for matching I.e pattern should be at the starting of the string**

**^S**

**^\d{3}**

**$ → matching pattern should be at the end**

**\d{3}$**

**Character classes**

**\d → matches any digits**

**\D → matches any non-digits**

**\s → matches white spaces**

**\w → matches any word (alpha-numeric)**

**[] → matches set of chars**

**T[abc]est-→ taest,tbest,tcest**

**T[t^abc]est →Tdest**

**\ → matches for actual ‘+’, ’.’ , ’\*’ chars**

**() → grouping of chars**

**([A-Z]\w+)**

**| → matches any one of the element**

**This (is|was|are|were) demo**

* **re is an internal package**

**1.findall()**

**seaerches the specified pattern in the string and return all matched string as list**

**listvariable=re.findall(‘pattern’,string)**

**2. search() → returns True if the specified pattern is in that string**

**otherwise False**

**variable=re.search(‘pattern’,string)**

**3.split() → split the string based on specified pattern and produce the output as list**

**variable=re.split(‘pattern’,string)**

**4.sub()-→ find the specified pattern and replace it with specified string**

**re.sub(“Pattern”,string to be replaced,source string)**

**psutil → https://psutil.readthedocs.io/en/latest/**

**API**