PYTHON REGULAR EXPRESSIONS

A *regular expression* is a special sequence of characters that helps you match or find other strings or sets of strings, using a specialized syntax held in a pattern. Regular expressions are widely used in UNIX world.

The module **re** provides full support for Perl-like regular expressions in Python. The re module raises the exception re.error if an error occurs while compiling or using a regular expression.

We would cover two important functions, which would be used to handle regular expressions. But a small thing first: There are various characters, which would have special meaning when they are used in regular expression. To avoid any confusion while dealing with regular expressions, we would use Raw Strings as **r'expression'**.

The *match* Function

This function attempts to match RE *pattern* to *string* with optional *flags*.

Here is the syntax for this function —

```
re.match(pattern, string, flags=0)
```

Here is the description of the parameters:

Parameter	Description
pattern	This is the regular expression to be matched.
string	This is the string, which would be searched to match the pattern at the beginning of string.
flags	You can specify different flags using bitwise OR |. These are modifiers, which are listed in the table below.

The *re.match* function returns a **match** object on success, **None** on failure. We use $group_{num}$ or *groups* function of **match** object to get matched expression.

Match Object Methods	Description
group$_{num}$ = 0	This method returns entire match $or specific subgroup num$
groups	This method returns all matching subgroups in a tuple $empty if there weren't any$

Example

```
#!/usr/bin/python
import re

line = "Cats are smarter than dogs"

matchObj = re.match( r'(.*) are (.*?) .*', line, re.M|re.I)

if matchObj:
   print "matchObj.group() : ", matchObj.group()
   print "matchObj.group(1) : ", matchObj.group(1)
   print "matchObj.group(2) : ", matchObj.group(2)
```