Regular expressions are used for pattern matching

to design regular expression we use following symbols

.	it matches with 1 character
[a-zA-Z]	it matches with any alphabet
[0-9] or \d	It matches with single digit
\w	it matches with any one word character [a-zA-Z0-9_]
\W	it matches with any one word character other than alphabet, digits and underscore [^a-zA-Z0-9_]
\D	any non digit character
\b	any one boundary character
\B	any one non boundary character
\s	it matches with space
\S	it matches with any non-space character
*	0 or more occurrences
+	1 or more occurrences
?	0 or 1 occurrence
{n}	exactly n occurences
{m,n}	minimum m occurrences, maximum n occurrences
{m,}	minimum m occurrences
^	it matches the pattern at the beginning of the string
$	it matches the pattern at the end of the string

s	[Oo]r	^[Oo]r	[Oo]r$	\b[Oo]r\b
This is origami	y	N	N	N
This is normal	y	N	N	N
There is a tailor	y	N	Y	N
this or that	y	N	n	Y
Or	y	y	Y	Y
There is a cat	N	N	N	N
Origami is good	y	Y	N	N

S=”Something is there somewhere”

[Ss].*e	Something is there somewhere		
[sS].*?e	Some		
\w+\s\w+\s\w+	Something is there somewhere		
^\w+\s\w+\s\w+$	This is line		

functions in re module

re.search(pattern,string,flags)	it will find the first occurrence of the given pattern, anywhere in the string, and returns a match object
re.match(pattern,string,flags)	it will find the first occurrence of the given pattern, at the begining in the string, and returns a match object
re.findall(pattern,string,flags)	It will find all occurrences of the pattern in the string and returns a list of strings
re.finditer(pattern,string,flags)	It will find all occurrences of the pattern in the string and returns a list of match objects
re.compile(pattern,flags)	It returns a regular expression object, and stores pattern and flags in it
re.sub(pattern, newstr,s,count=0,flags=re.I|re.M)	It will substitute all occurrences of the given pattern, by newstr but if count is given , then it will replacecount number of occurrences with the given string

file handling

fh=open("mytetx.txt")	open the file in read mode and returns the handle
fh1=open("mytetx.txt","w")	open the file in write mode and returns the handle, if file exists it will overwrite the file, otherwise it creates the files
fh.close()	I closes the file
fh.readlines()	will read all lines from the file and store it in a list
fh.read()	will read all data from the file and store it in a string
fh.read(n)	it reads n characters from current position
fh.seek(n)	moves the poiter at n th position
fh.tell()	gives the current pointer position