Regular Expression

- First we have to import regular expression package **re**
- **re.methodname(pattern/pattern varible,String/string variable)**

Methods in re

- search()
- match()
- findall()

In [3]:
```
import math

math.sqrt(36)
```

Out[3]:
```
6.0
```

In [4]:
```
36**0.5
```

Out[4]:
```
6.0
```

In [14]:
```
import re

print(re.search("ss","ApssdcssDssDss"))
print(re.search("Pyt","APSSDC"))
```

```
<re.Match object; span=(2, 4), match='ss'>
None
```

In [9]:
```
n = input()
print(re.search("@",n))
```

```
APSSDC@1521
<re.Match object; span=(6, 7), match='@'>
```

In [16]:
```
# Match method
print(re.match("As","Apssdc"))  # check charecter by charecter
print(re.match("APS","APSSDC"))

```

...

In [19]:
```
# Findall
print(re.findall("ss","ApssdcssDssDss"))
s = re.findall("ss","ApssdcssDssDss")
len(s)
```

```
['ss', 'ss', 'ss', 'ss']
```

Out[19]: 4

In [20]:
```
# 7013021421  match ("701","7013021421")  matching
# 0123456789  search ("7")
```

Symbols in Re

Character	Description	Example
[]	A set of characters	"[a-m]"
\	Signals a special sequence (can also be used to escape special characters)	"\d"
.	Any character (except newline character)	"he..o"
^	Starts with	"^hello"
$	Ends with	"world$"
*	Zero or more occurrences	"aix*"
+	One or more occurrences	"aix+"
{}	Exactly the specified number of occurrences	"al{2}"
|	Either or	"falls|stays"
()	Capture and group	

In [24]:
```
# "." Symbol
print(re.search("..","@APSSDC"))
print(re.search("..","AP"))
print(re.search("..","A"))
print(re.search("..",""))
```

```
<re.Match object; span=(0, 2), match='@A'>
<re.Match object; span=(0, 2), match='AP'>
<re.Match object; span=(0, 1), match='A'>
None
```

In [25]:
```
# "^" symbol

print(re.search("^A","APSSDC"))
print(re.search("^AP","APSSDC"))
print(re.search("^A","SAPSSDC"))

```

```
<re.Match object; span=(0, 1), match='A'>
<re.Match object; span=(0, 2), match='AP'>
None
```

In [27]:
```
# "$" symbol
print(re.search("DC$","APSSDC"))
print(re.search("C$","APSSDC"))
print(re.search("DC$","APSSDS"))
print(re.match("DC$","APSSDC"))


```

```
<re.Match object; span=(4, 6), match='DC'>
<re.Match object; span=(5, 6), match='C'>
None
None
```

In [31]:
```
# "*"  symbol

print(re.search("A*","AAAAAAAPSSDC"))
print(re.search("A*","PSSDC"))
print(re.search("A","PSSDC"))

```

```
<re.Match object; span=(0, 7), match='AAAAAAA'>
<re.Match object; span=(0, 0), match=''>
None
```

In [32]:
```
# "+" symbol  # minimum one time

print(re.search("A+","AAAAAAAPSSDC"))
print(re.search("A+","PSSDC"))
print(re.search("A+","APSSDC"))

```

```
<re.Match object; span=(0, 7), match='AAAAAAA'>
None
<re.Match object; span=(0, 1), match='A'>
```

In [39]:

```
# {min,max}
print(re.search("A{1,5}","AAAAAAAPSSDC"))
print(re.search("A{1,3}","PSSDC"))
print(re.search("A{0,1}","PSSDC"))

```

```
<re.Match object; span=(0, 5), match='AAAAA'>
None
<re.Match object; span=(0, 0), match=''>
```

In [44]:

```
# []

print(re.search("[ST]","APSSDC"))
print(re.search("[TV]","PSSTDC"))
print(re.search("TV","APSSTDC"))
print(re.match("[ASP]","PSSDC"))
print(re.match("[TV]","APSSDCTV"))

```

```
<re.Match object; span=(2, 3), match='S'>
<re.Match object; span=(3, 4), match='T'>
None
<re.Match object; span=(0, 1), match='P'>
None
```

In [47]:

```
# "\d,\D,\s,\S"
print(re.search("\d","AA12AAAAAPSSDC"))
print(re.search("\d\d","PSS12DC"))
print(re.search("\D","12PSSDC")) #other than digit
print(re.match("\d\d","12AAAAAAAPSSDC"))
print(re.match("\d\d","Ss12AAAAAAAPSSDC"))

```

```
<re.Match object; span=(2, 3), match='1'>
<re.Match object; span=(3, 5), match='12'>
<re.Match object; span=(2, 3), match='P'>
<re.Match object; span=(0, 2), match='12'>
None
```

In [ ]:

```

```