**Regular expressions**

We have to import the package re

**Methods**

- search
- match
- findall

**syntax: re.methodname(pattern,string)**

In [3]:
```
import math

math.sqrt(4)
math.log2(10)
```

Out[3]:
```
3.321928094887362
```

In [8]:
```
import re

## search

print(re.search("SD","APSSDC"))
print(re.search("SA","APSSDC"))
print(re.search("PSSD","APSSDC"))

```

```
<re.Match object; span=(3, 5), match='SD'>
None
<re.Match object; span=(1, 5), match='PSSD'>
```

In [10]:
```
# match
print(re.match("SD","APSSDC"))
print(re.match("A","APSSDC"))
print(re.match("APC","APSSDC"))
```

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

In [13]:
```
# find all

print(re.search("SD","APSSDSDSDSDSDC"))
s  = re.findall("SD","APSSDSDSDSDSDC")
len(s)
```

```
<re.Match object; span=(3, 5), match='SD'>
```

Out[13]:
```
5
```

### Symbols

| 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 [20]:
```
# . Symbol

print(re.search("..","APSSDC"))
print(re.search("..","A"))
print(re.search("..","   APSSDC"))
print(re.search("..","  "))
print(re.search("A..","APSSDC"))
print(re.search("S..","APSS"))
```

```
<re.Match object; span=(0, 2), match='AP'>
None
<re.Match object; span=(0, 2), match='  '>
None
<re.Match object; span=(0, 3), match='APS'>
None
```

In [26]:
```
print(re.search("^AP..","APSSDC"))
print(re.search("^A","PSSADC"))
print(re.search("A..","PSSADC"))

```

```
<re.Match object; span=(0, 4), match='APSS'>
None
<re.Match object; span=(3, 6), match='ADC'>
```

In [28]:
```
print(re.match("^AP..","APSSDC"))
print(re.match("^A","PSSADC"))
print(re.match("AP..","PSSADC"))

```

```
<re.Match object; span=(0, 4), match='APSS'>
None
None
```

In [ ]:

```
1
```