Name validator

- M Srilalitha
- M Sri Lalitha
- Mulpuru Srilalitha
- Mulpuru Sri Lalitha
- M. Srilalitha
- Srilalitha M

In [1]:
```
1 import re
2 patt = "[A-Z][A-Za-z]{0,10}[.]{0,1}[ ][A-Z][A-Za-z]{2,10}[ ]{0,1}[A-Z]{0,1}[
3 name = "M Srilalitha"
4 re.match(patt,name)
```

Out[1]:
```
<re.Match object; span=(0, 12), match='M Srilalitha'>
```

Email Validator

- srilalitha.m@apssdc.in (mailto:srilalitha.m@apssdc.in)
- All letters including starting letter must be lowercase alphabet
- Cotains some special charecters (optional)
- contains some numbers(optional)
- contains some digits or alphabets after special charecter
- Must contains @
- Collection of alphabets ===> len 4-8
- must contain .
- Collection of alphabets ===> len 2-4

In [3]:
```
1 pat = "[a-z][a-z0-9]{5,13}[.-_#]{0,1}[a-z0-9]{1,5}[@][a-z]{4,8}[.][a-z]{2,4}
2 mail ="srilalitha.m@apssdc.in"
3 re.match(pat,mail)
```

Out[3]:
```
<re.Match object; span=(0, 22), match='srilalitha.m@apssdc.in'>
```

Functions

- code reusability

Syntax:

define a funtcion() using def keyword: -set of statements/logic function calling()

In [14]:
```
def add():  # parameters  # retrun
    a = 10
    b = 5
    c= a+b
    print(c)
add() # arguments
```

```
15
```

In [12]:
```
def even(n):
    if n%2==0:
        print("even")
    else:
        print("odd")
n = int(input())
even(n)
```

```
10
even
```

Types of functions

- without argument and without return value
- without argument and with return value
- with argument and without return value
- with argument and with return value

In [6]:
```
# without argument and with return value

def sub():
    a = 10
    b =5
    print(a-b)  #return statement
sub()
```

```
5
```

In [21]:
```
mul()
```

Out[21]:
```
50
```

In [4]:
```
# without argument and with return value

def mul():
    a = 10
    b =5
    c = a*b
    return c   #return statement
mul()
```

Out[4]: 50

In [20]:
```
mul()
```

Out[20]: 50

In [9]:
```
# with arguments and without return value

def multiplication(a,b):
    print(a*b)
#a = int(input())
#b = int(input())
multiplication(2,3)
```

6

In [10]:
```
# with arguments and with return value

def div(a,b):
    d = a//b
    return d
div(4,2)
```

Out[10]: 2

In [13]:
```
for i in range(1,10):
    even(i)

```

...

In [15]:
```
def fact(n):
    fact = 1 #1,2,6
    for i in range(1,n+1):  # 1,2,3
        fact = fact*i
    print(fact)
fact(5)
```

120

In [16]:
```
for i in range(1,10):
    fact(i)

```

...

In [23]:
```
def prime(n):
    c =0
    for i in range(1,n+1):
        if n%i==0:
            c+=1
    if c==2:
        print(i, end = " ")
prime(7)
```

```
7
```

In [24]:
```
for i in range(1,100):
    prime(i)
```

```
2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97
```

Types of arguments

- formal arguments
- Actual arguments
  - Position argument
  - keyword
  - defalut
  - variable length

In [25]:
```
def add(a,b):  # formal arguments
    c = a+b
    print(c)
add(2,3) # actual
```

```
5
```

In [30]:
```
# position arguments

def person(name,age):
    print(name)
    print(age)
name = input()
age = int(input())
person(name,age)
```

...

In [29]:
```
1 person(name,age)
```

```
lalitha
25
```

In [32]:
```
1 # keyword
2
3 def person(name,age):
4     print("name:",name)
5     print("age:",age)
6
7 person(age = 30, name ="xyz")
```

```
name: xyz
age: 30
```

In [33]:
```
1 # default
2 def person(name,age=20):
3     print("name:",name)
4     print("age:",age)
5
6 person("xyz")
```

```
name: xyz
age: 20
```

In [34]:
```
1 # variable length arguments
2
3 def person(name,age):
4     print("name:",name)
5     print("age:",age)
6
7 person("xyz",30,24,21)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-34-d652fdc2d128> in <module>
      5     print("age:",age)
      6
----> 7 person("xyz",30,24,21)

TypeError: person() takes 2 positional arguments but 4 were given
```

In [37]:

```
def person(name,age1,*age):
    print("name:",name)
    print("age1:",age1)
    print("age:",age)

person("xyz",30,24,21,67,89)
```

```
name: xyz
age1: 30
age: (24, 21, 67, 89)
```

In [ ]:

```

```