Finding Substrings:
In PYTHON programming to find sub strings we can use the following 4 methods:
Forward direction:
1 find() 2 index()
Backward direction:
1 rfind() 2 rindex()

find():
Returns index of first occurrence of the given substring. If it is not available then we will get -1

Syntax:
String.find(substring,begin,end)

Example:
```
PyStr="Learning Python is Simpler"
print(PyStr.find("Python"))
print(PyStr.find("Data"))
print(PyStr.find("e"))
```

Example:
```
PyStr="Python is"
#String.find(substring,begin,end)
print(PyStr.find('i'))
print(PyStr.find('s'))
PyStr="Python is good one"
print(PyStr.find('o'))
print(PyStr.find('o',4))
print(PyStr.find('o',5))
print(PyStr.find('o',12,16))
print(PyStr.find('o',13,16))
print(PyStr.find('O',13,16))
```

Example:
```
PyStr="hellopythonisgreat"
print(PyStr.find('a'))
print(PyStr.find('b',7,15))
print(PyStr.find('t',7,15))
print(PyStr.find('t',8,15))
```

index() method:
It returns the index of a substring inside the string (if found). If the substring is not found, it raises an exception.

Syntax:
str.index(sub[, start[, end]])

Example:
```
PyStr='Python programming is fun'
print(PyStr.index('is fun'))
print(PyStr.index('ing', 10))
print(PyStr.index('g is', 10, -4))
```

Backward direction:

1 rfind() 2 rindex()

rfind()
It returns the highest index of the substring (if found). If not found, it returns -1.

Syntax:
```
str.rfind(sub[, start[, end]] )
```

Example:
```
PyStr="Learning Python is Simpler"
print(PyStr.rfind("S"))
print(PyStr.rfind("e"))
print(PyStr.rfind("o",10))
print(PyStr.rfind("e",15,25))
```

rindex()
It returns the highest index of the substring inside the string (if found). If the substring is not found, it raises an exception.

Syntax:
```
str.rindex(sub[, start[, end]] )
```

Example:
```
PyStr='Do small things with great love'
print(PyStr.rindex('u'))
print(PyStr.rindex('t', 2))
print(PyStr.rindex('h', 6, 20))
```

count()
It returns the number of occurrences of a substring in the given string.

Syntax:
```
string.count(sub[, start[, end]])
```

Example
```
PyStr = "Python is Awesome, Yes or Not"
print(PyStr.count('i'))
print(PyStr.count('o',1))
print(PyStr.count('s',10,25))
```

startswith()
It returns True if a string starts with the specified prefix(string). If not, it returns False.

Syntax:
```
str.startswith(prefix[, start[, end]])
```

Example:
```
PyTxt="Python programming is easy."
print(PyTxt.startswith('programming is', 7))
print(PyTxt.startswith('programming is', 7, 18))
print(PyTxt.startswith('programming is', 7, 21))
```

split():
It splits a string into a list.

Syntax:
string.split(separator, max)

Example:
```
PyStr="hello my name is Raju"
print(PyStr.split(" "))
```
O/P:
```
['hello', 'my', 'name', 'is', 'Raju']
```

Example:
```
PyStr="hello#my#name is Raju"
print(PyStr.split("#"))
```
O/P:
```
['hello', 'my', 'name is Raju']
```

Example:
```
PyStr="hello#my#name is Raju"
print(PyStr.split("#",1))
```
O/P:
```
['hello', 'my'#'name is Raju']
```

Example:
```
PyStr="Hello-Welcome-To-PYTHON"
Str=PyStr.split('-')
for x in Str:
    print(x)
```

Example:Reading Multiple Inputs:
```
>>> x=input().split()
10 20 30
>>> x
['10', '20', '30']
```