within triple quotes
within triple single quotes or double quotes, we represent multiline string. Triple quotes in python also used to represent doc string

```
def power()
‘’’ power is user defined function
This function is used to find power of a number’’’
```

Example:

```
>>> str1='''python is
... general purpose
... programming
... language'''
>>> print(str1)
python is
general purpose
programming
language
>>> str2="""python is
... multiparadigm
... programming
... language"""
>>> print(str2)
python is
multiparadigm
programming
language
```

Escape Sequences or backslash character values/literals
These are special sequences which are used inside string

Escape Sequences	**Description**
\n	Newline
\t	Horizontal tab space
\v	Vertical tab space
\\	Backslash
\’	‘
\”	“

```
>>> str2='python is a 'scripting' programming language'
...
SyntaxError: invalid syntax
```

```
>>> str2='python is a \'scripting\' langauge'
...
>>> print(str2)
...
python is a 'scripting' langauge
>>> str3="python is a "scripting" langauge"
...
SyntaxError: invalid syntax
>>> str3="python is a \"scripting\" language"
...
>>> print(str3)
...
python is a "scripting" language

>>> str4="python\tjava\toracle"
...
>>> print(str4)
...
python       java   oracle
```

int
float
complex
bool
NoneType
Str

Rollno=1 → int
Name=”naresh” → string
Course=”python” → string
Fee=2000.0 → float
Fee_paid=True → Boolean

Every program required 3 things

1. Input

2. Process
3. Output

Output

Output is information or result given by program
This output displayed on console/monitor
Output required dest (file, database, console, printer, …)

print() function

print() is a standard output function
this function is used to print data or information on console or monitor.
This function is available in one library __builtins__

```
C                                   Python
--                                  ----------
#include<stdio.h>                   print(“Hello”)
void main()
{
printf(“Hello”);
}
```

__builtins__ default library imported by any python

```
>>> dir(__builtins__)

['ArithmeticError', 'AssertionError', 'AttributeError', 'BaseException',
'BlockingIOError', 'BrokenPipeError', 'BufferError', 'BytesWarning',
'ChildProcessError', 'ConnectionAbortedError', 'ConnectionError',
'ConnectionRefusedError', 'ConnectionResetError', 'DeprecationWarning',
'EOFError', 'Ellipsis', 'EncodingWarning', 'EnvironmentError', 'Exception',
'False', 'FileExistsError', 'FileNotFoundError', 'FloatingPointError',
'FutureWarning', 'GeneratorExit', 'IOError', 'ImportError', 'ImportWarning',
'IndentationError', 'IndexError', 'InterruptedError', 'IsADirectoryError',
'KeyError', 'KeyboardInterrupt', 'LookupError', 'MemoryError',
'ModuleNotFoundError', 'NameError', 'None', 'NotADirectoryError',
'NotImplemented', 'NotImplementedError', 'OSError', 'OverflowError',
'PendingDeprecationWarning', 'PermissionError', 'ProcessLookupError',
'RecursionError', 'ReferenceError', 'ResourceWarning', 'RuntimeError',
'RuntimeWarning', 'StopAsyncIteration', 'StopIteration', 'SyntaxError',
'SyntaxWarning', 'SystemError', 'SystemExit', 'TabError', 'TimeoutError',
'True', 'TypeError', 'UnboundLocalError', 'UnicodeDecodeError',
'UnicodeEncodeError', 'UnicodeError', 'UnicodeTranslateError',
'UnicodeWarning', 'UserWarning', 'ValueError', 'Warning', 'WindowsError',
'ZeroDivisionError', '_', '__build_class__', '__debug__', '__doc__',
'__import__', '__loader__', '__name__', '__package__', '__spec__', 'abs',
'aiter', 'all', 'anext', 'any', 'ascii', 'bin', 'bool', 'breakpoint', 'bytearray', 'bytes',
'callable', 'chr', 'classmethod', 'compile', 'complex', 'copyright', 'credits',
'delattr', 'dict', 'dir', 'divmod', 'enumerate', 'eval', 'exec', 'exit', 'filter', 'float',
'format', 'frozenset', 'getattr', 'globals', 'hasattr', 'hash', 'help', 'hex', 'id',
'input', 'int', 'isinstance', 'issubclass', 'iter', 'len', 'license', 'list', 'locals', 'map',
'max', 'memoryview', 'min', 'next', 'object', 'oct', 'open', 'ord', 'pow', 'print',
'property', 'quit', 'range', 'repr', 'reversed', 'round', 'set', 'setattr', 'slice',
'sorted', 'staticmethod', 'str', 'sum', 'super', 'tuple', 'type', 'vars', 'zip']
```

Syntax:

```
print(values,sep=’ ‘,end=’\n’,file=sys.stdout)
```

1. values
2. Sep
3. End
4. File

values: print function receive 0 more values and combine all these values into string.

```
>>> print(10)

10
>>> print(10,20)

10 20
>>> print(10,20,30,40,50)

10 20 30 40 50
>>> print(10,20,30,40,50,sep="*")

10*20*30*40*50
```

```
>>> print(101,"naresh","python")
...
101 naresh python
>>> print(101,"naresh","python",sep=":")
...
101:naresh:python
>>> print(10,20,30,40,50,sep="\n")
...
10
20
30
40
50
>>> print(10,20,30,40,50,sep="\t")
...
10      20      30      40      50
>>> print(10,sep="*")
...
10
```