print() in python

print() in python is used to print your message or value of the variable on the output screen.

Syntax:

print(object(s), sep=separator, end=end, file=file) object(s) Any object, and as many as you like. sep= 'separator' Optional. Specify how to separate the objects end='end' Optional. Specify what to print at the end. Default is '\n' file Optional. An object with a write method. Default is sys.stdout which is your output screen

- Print a message
- Print one or more variable(s)
- Print any object

In [1]:
```
print('Hello world')
print('Input', 'output', 'functions',sep = "-")
print('Welcome to the',end = " + ")
print("30 days of programming")
```

```
Hello world
Input-output-functions
Welcome to the + 30 days of programming
```

File option in python

(optional) We will learn this method in FILE HANDLING.

In [4]:
```
f = open('python.txt', 'w')                # Created a file
```

```
print('Pretty cool, huh!', file = f)    # Write in the file
f.close()

f = open('python.txt')                  # open the file as read mode
print(f.read())                         # read the content of the file
 and print it
```

```
Pretty cool, huh!
```

Different ways to print the values of variable in python

In [5]:
```
x = 2
y = 'Hi'
```

In [7]:
```
print("x=",x,'y=', y)            ## Note ','(coma) in python automatica
lly adds an addtional space while printing.
print("x= {0} y= {1}".format(x,y))
print("x= %d y= %s"%(x,y))
```

```
x= 2 y= Hi
x= 2 y= Hi
x= 2 y= Hi
```

In the last print method we have used %d and %s.
What is this?
These are known as format specifiers.
Since x contains an integer value, so x is of integer type and thus we used %d.
Similarly, y is of string type, and thus we use %s.

Data type	Format specifier
int	%d
float	%f or %g
string	%s

You can check the type of the varibale using type()

```
In [8]: x = 15.545353
```

```
In [9]: type(x)
```

```
Out[9]: float
```

```
In [10]: print("%.3f"%x)                  # here .3 defines 3 digits after decimal
         print("{:.3}".format(x))         # here .3 defines the no of digits from the
          right
```

```
15.545
15.5
```

Input() in python

In python we can assign value to the variable by user at run time

```
In [11]: a = input()
         b = input("Enter input")
```

```
4
Enter input3
```

Note:

The type of the varible which os assigned by input will always be string.

```
In [12]: type(a)
```

```
Out[12]: str
```

So, we have to convert this as of our need.

```
In [13]: a = int(input('Enter no. '))
         b = float(input('Enter no. '))
         print('Type of a',type(a))
         print('Type of b',type(b))
```

```
Enter no. 4
Enter no. 5
Type of a <class 'int'>
Type of b <class 'float'>
```

```
In [14]: a = int(input("Enter data"))
```

```
Enter datastring

---------------------------------------------------------------------------
----
ValueError                                Traceback (most recent call l
ast)
<ipython-input-14-4e62f22a1473> in <module>
----> 1 a = int(input("Enter data"))

ValueError: invalid literal for int() with base 10: 'string'
```

```
In [ ]:
```