## Chapter - 6 : Functions

## Built-in Python Functions

| SNo | Function | Description |
|-----|----------|-------------|
| 13. | abs() | Returns the absolute value of a number |
| 14. | eval() | Evaluates and executes an expression |
| 15. | round() | Rounds a numbers |
| 16. | max() | Returns the largest item in an iterable |
| 17. | min() | Returns the smallest item in an iterable |
| 18. | oct() | Converts a number into an octal |
| 19. | hex() | Converts a number into an hexadecimal |
| 20. | pow() | Returns the value of x to the power of y |
| 21. | range() | Returns a sequence of numbers, starting from 0 and increments by 1 (by default) |

### 13. abs() Function

This function returns the absolute value of the specified number.

```
x = abs(-7.25)
print(x)                          Output : 7.25
x = abs(3+5j)
print(x)                          Output : 5.830951894845301
```

### 14. eval() Function

This function evaluates the specified expression, if the expression is a legal Python statement, it will be executed.

```
x= 'print(55)'
eval(x)
x= eval( '10+5' )
print(x)
```

**Output-**          55
                     5

### 15. round() Function

- This function returns a floating point number that is a rounded version of the specified number, with the specified number of decimals.
- The default number of decimals is 0, meaning that the function will return the nearest integer.

Syntax :  round(number, digits)

number        - Required. The number to be rounded
digits          - Optional. The number of decimals upto rounded. Default is 0

```
x = round(15.76543 , 2)
print(x)
x = round(15.76543)
print(x)
x = round(15.76543 , -1)
print(x)
```

**Output-**        15.77
16
20.0

### 16. max() Function

- This function returns highest value from the items or an iterable.
- If the values are strings, an alphabetically comparison is done.

Syntax :         max( n1, n2, n3, ...)
or
max(iterable)

```
x = max(5, 10)
print(x)
x = max("Mike", "John", "Vicky")
print(x)
a=(1, 5, 3, 9)
x = max(a)
print(x)
a={"Mike":1, "John":2, "Vicky":3}
x = max(a)
print(x)
```

**Output**  :         10
Vicky
9
Vicky

## 17. min() Function :

- This function returns lowest value from the items or an iterable.
- If the values are strings, an alphabetically comparison is done.

Syntax :  min( n1, n2, n3, ...)
or
min(iterable)

```
x = min(5, 10)
print(x)
x = min("Mike", "John", "Vicky")
print(x)
a=(1, 5, 3, 9)
x = min(a)
print(x)
a={"Mike":1, "John":2, "Vicky":3}
x = min(a)
print(x)
```

**Output** :     5
John
1
John

## 18. oct() Function :

The oct() function converts an integer into an octal string.
Octal strings in Python are prefixed with 0o (Zero followed by "o")

```
x = oct(12)
print(x)
```
**Output** –     0o14

## 19. hex() Function

The hex() function converts an integer into an hexadecimal string.
Hexadecimal strings in Python are prefixed with 0x (Zero followed by "x")

```
x = hex(12)
print(x)
```
**Output** –     0xc

## 20. pow() Function

This function returns the value of **x** to the power of **y** ($x^y$).
If a third parameter is present, it returns x to the power of y, modulus z.

Syntax        pow(x, y, z)
                    where  x  - base , y –exponent , z – modulus
                    z is the optional.

```
x = pow(4, 3)          #  4 * 4 * 4
print(x)
x = pow(4, 3, 5)       #  (4 * 4 * 4) % 5
print(x)
```


## 21. range() Function

This function returns a sequence of numbers, starting from 0 by default, and increments by 1 (by default), and ends at a specified number.

**Syntax**        range*(start, stop, step)*
start    Optional. An integer number specifying at which position to start. Default is 0
stop    Required. An integer number specifying at which position to end.
step    Optional. An integer number specifying the incrementation. Default is 1

**#a sequence of numbers from 0 to 5**
```
x = range(6)
for n in x:
   print(n)
```

**#a sequence of numbers from 3 to 5**
```
x = range(3, 6)
for n in x:
    print(n)
```

**#a sequence of numbers from 3 to 19, but increment by 2**
```
x = range(3, 20, 2)
for n in x:
   print(n)
```