Task 1:

Make a mini-calculator using functions. All the functions should be accessed from module. Your program should ask inputs and option from user.

Task 2:

Write a function which can apply this function in all elements of list. Y=x √

using list

comprehension.

Task 3:

Apply all arithmetic functions on Numpy array and discus output.

Task 4:

Make a Numpy array by using all possible ways.

Task 5:

Perform basic indexing and slicing on multi-dimension array.

Task 1:

Make a mini-calculator using functions. All the functions should be accessed from module. Your program should ask inputs and option from user.

In [4]:

```
def add(a , b):
    return a+b

def sub(a ,b):
    return a-b

def mul(a , b):
    return a * b

def div(a,b):
    return a/b

def squared(a,b):
    return a ** b



print("mul",mul(5,2))
print("add",add(5,2))
print("sub",sub(5,2))
print("div",div(5,2))
print("Squared ",squared(5,2))
```

```
mul 10
add 7
sub 3
div 2.5
Squared  25
```

Task 2:

```
Write a function which can apply this function in all elements of list. Y=x √

using list
```

In [7]:

```
from math import sqrt
def fun(x):

    return (sqrt(x)-(4*x))/x

fun(6)
```

Out[7]:

```
-3.591751709536137
```

Task 3:

Apply all arithmetic functions on Numpy array and discus output.

In [34]:

```
a1 = np.array([1,2,3])
a2 = np.random.randint(0,10,size=(2,3))

a1,a2
```

Out[34]:

```
(array([1, 2, 3]),
 array([[5, 9, 7],
        [0, 5, 3]]))
```

In [28]:

```
ones = np.ones(3)
ones
```

Out[28]:

```
array([1., 1., 1.])
```

In [29]:

```
a1 + ones
```

Out[29]:

```
array([2., 3., 4.])
```

In [30]:

```
a1 - ones
```

Out[30]:

```
array([0., 1., 2.])
```

```
In [31]: a1 * ones

Out[31]: array([1., 2., 3.])

In [32]: a1 * 5

Out[32]: array([ 5, 10, 15])

In [35]: a2

Out[35]: array([[5, 9, 7],
                [0, 5, 3]])

In [36]: a1 ** a2

Out[36]: array([[   1,  512, 2187],
                [   1,   32,   27]])

In [37]: a1 - a2

Out[37]: array([[-4, -7, -4],
                [ 1, -3,  0]])
```

Task 4:

Make a Numpy array by using all possible ways.

```
In [26]: a1 = np.array([1,2,3])
         a1

Out[26]: array([1, 2, 3])
```

In [27]:
```
a2 = np.array([
    [
        [1,2,3],
        [1,5,6],
        [5,6,7]
    ],
    [
        [4,5,6],
        [1,3,5],
        [1,4,5]
    ]
])
a2
```

Out[27]:
```
array([[[1, 2, 3],
        [1, 5, 6],
        [5, 6, 7]],

       [[4, 5, 6],
        [1, 3, 5],
        [1, 4, 5]]])
```

In [11]:
```
ones = np.ones((2,3))
ones
```

Out[11]:
```
array([[1., 1., 1.],
       [1., 1., 1.]])
```

In [12]:
```
np.zeros((2,3))
```

Out[12]:
```
array([[0., 0., 0.],
       [0., 0., 0.]])
```

In [13]:
```
range_array = np.arange(0,11,2 )
range_array
```

Out[13]:
```
array([ 0,  2,  4,  6,  8, 10])
```

```
In [15]: random_array = np.random.randint(0,10,size=(2,3,5))
         random_array

Out[15]: array([[[7, 2, 8, 4, 7],
                 [7, 7, 3, 7, 3],
                 [4, 4, 4, 5, 0]],

                [[6, 9, 5, 7, 2],
                 [8, 1, 4, 1, 2],
                 [5, 2, 1, 9, 3]]])
```

```
In [19]: rand = np.random.random((1,1,15,5))
         rand

Out[19]: array([[[[0.84609716, 0.99360092, 0.34239463, 0.052884  , 0.97938858]]]])
```

```
In [20]:

Out[20]: array([[[0.41370797, 0.08976311],
                 [0.22240868, 0.22704704],
                 [0.78363215, 0.9178051 ]],

                [[0.38388878, 0.32795903],
                 [0.79027016, 0.0538381 ],
                 [0.42007477, 0.68192605]]])
```

Task 5:

Perform basic indexing and slicing on multi-dimension array.

```
In [21]: arr = np.unique(np.arange(10))
         arr

Out[21]: array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
```

```
In [22]: arr[::-2]

Out[22]: array([9, 7, 5, 3, 1])
```

```
In [23]: arr3 = np.random.randint(20,size=(5,3,4))
         arr3

Out[23]: array([[[10,  2, 13,  1],
                 [ 5, 11, 15, 14],
                 [ 4,  8, 12, 15]],

                [[19,  4,  3, 16],
                 [18,  9,  1,  2],
                 [ 0, 13, 12, 11]],

                [[19, 11,  6,  1],
                 [ 4, 13,  9, 19],
                 [15,  6, 19,  3]],

                [[ 7,  7,  8, 13],
                 [ 8,  8, 14, 19],
                 [18, 13,  4, 16]],

                [[18,  6, 16,  2],
                 [ 3,  1, 12, 13],
                 [11, 15,  1, 15]]])
```

```
In [24]: arr3[0,:2,:3]

Out[24]: array([[10,  2, 13],
                [ 5, 11, 15]])
```

```
In [ ]: 
```