In [98]:

```
#Task 1:
```

1: Write a function to compute 5/0 and use try/except to catch the exceptions.

In [106]:

```
def divide(a,b):
    try:
        return a/b
    except Exception as e:
        print('Exception occured : {}'.format(e))
```

In [116]:

```
print(divide(5,1))
```

```
5.0
```

In [117]:

```
print(divide(5,0))
```

```
Exception occured : division by zero
None
```

2: Implement a Python program to generate all sentences where subject is in ["Americans","Indians"] and verb is in ["Play", "watch"] and the object is in ["Baseball","cricket"]

Output should come as below: Americans play Baseball. Americans play Cricket. Americans watch Baseball. Americans watch Cricket. Indians play Baseball. Indians play Cricket. Indians watch Baseball. Indians watch Cricket.

In [118]:

```
subjects=["Americans","Indians"]
verbs   =["play","watch"]
objects= ["Baseball","Cricket"]

for i in subjects:
    for j in verbs:
        for k in objects:
            print('{} {} {}'.format(i,j,k))
```

```
Americans play Baseball
Americans play Cricket
Americans watch Baseball
Americans watch Cricket
Indians play Baseball
Indians play Cricket
Indians watch Baseball
Indians watch Cricket
```

In [120]:

```
#Task 2:
```

1:Write a function so that the columns of the output matrix are powers of the input vector.The order of the powers is determined by the increasing boolean argument. Specifically, when increasing is False, the i-th output column is the input vector raised element-wise to the power of N - i - 1.

In [122]:

```
import numpy as np
```

In [126]:

```
def myvander(x,N,flag):
        if flag =='True':
            z=np.array([x**(N-i-1) for i in range(N-1,-1,-1) ])
        elif flag =='False':
            z=np.array([x**(N-i-1) for i in range(N) ])
        else:
            z=np.array([x**(N-i-1) for i in range(N) ])
        return(np.transpose(z))
```

In [127]:

```
x = np.array([1,2,3,4])
N=5
```

In [128]:

```
### My using myvander()
v = myvander(x,N,'True')

print('Alexandre-Theophile Vandermonde produced by myvander() function\n {}'.f
ormat(v))
```

```
Alexandre-Theophile Vandermonde produced by myvander() function
 [[  1   1   1   1   1]
 [  1   2   4   8  16]
 [  1   3   9  27  81]
 [  1   4  16  64 256]]
```

In [130]:

```
### My using numpy vander()
v1=np.vander(x,N,increasing=True)

print('Alexandre-Theophile Vandermonde produced by built in numpy function\n
{}'.format(v1))
```

```
Alexandre-Theophile Vandermonde produced by built in numpy function
 [[  1   1   1   1   1]
 [  1   2   4   8  16]
 [  1   3   9  27  81]
 [  1   4  16  64 256]]
```

In [131]:

```
### My using myvander()
v = myvander(x,N,'False')

print('Alexandre-Theophile Vandermonde produced by myvander() function\n {}'.format(v))
```

```
Alexandre-Theophile Vandermonde produced by myvander() function
 [[  1   1   1   1   1]
 [ 16   8   4   2   1]
 [ 81  27   9   3   1]
 [256  64  16   4   1]]
```

In [132]:

```
### My using numpy vander()
v1=np.vander(x,N,increasing=False)

print('Alexandre-Theophile Vandermonde produced by built in numpy function\n {}'.format(v1))
```

```
Alexandre-Theophile Vandermonde produced by built in numpy function
 [[  1   1   1   1   1]
 [ 16   8   4   2   1]
 [ 81  27   9   3   1]
 [256  64  16   4   1]]
```