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

```
n=range(-10,10)
count=0
print("n=",n)
print("i   5 by i\n")
try:
    for i in n[::-1]:
        print(i,5/i)

except:
    print ("Error occurred at i= ",count)
```

```
n= range(-10, 10)
i   5 by i

9 0.5555555555555556
8 0.625
7 0.7142857142857143
6 0.8333333333333334
5 1.0
4 1.25
3 1.6666666666666667
2 2.5
1 5.0
Error occurred at i=  0
```

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”].

- Hint: Subject,Verb and Object should be declared in the program as shown below.
- subjects=[“Americans”,“Indians”] verbs=[“play”,“watch”] objects=[“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.

```
class players:
    def __init__(self,nation,game,verb):
            self._nation=nation
            self._game=game
            self._verb=verb

    def __str__(self):
        return "%s  %s %s \n"%(self._nation,self._verb,self._game)
```

```
p1=players("Americans","Baseball","play")
p2=players("Americans","Cricket","play")
p3=players("Americans","Baseball","watch")
p4=players("Americans","cricket","plays")
q1=players("Indians","Baseball","play")
q2=players("Indians","Cricket","play")
q3=players("Indians","Baseball","watch")
q4=players("Indians","cricket","plays")







print("\n",p1,"\n",p2,"\n",p3,"\n",p4,"\n",q1,"\n",q2,"\n",q3,"\n",q4)
```

```
 Americans  play Baseball

 Americans  play Cricket

 Americans  watch Baseball

 Americans  plays cricket

 Indians  play Baseball

 Indians  play Cricket

 Indians  watch Baseball

 Indians  plays cricket
```

Task 2:Write a function so that the columns of the output matrix are powers of the input vector.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.

- HINT: Such a matrix with a geometric progression in each row is named for Alexandre- Theophile Vandermonde.

URLS referred

- https://stackoverflow.com/questions/58868529/how-would-i-create-a-matrix-in-python-without-importing-modules-how-would-i-mak
- https://stackoverflow.com/questions/58868529/how-would-i-create-a-matrix-in-python-without-importing-modules-how-would-i-mak
- https://www.wikiwand.com/en/Vandermonde_matrix

Elements of a Vander Monde matrix is given by $V_{ij} = \alpha_i^{j-1}$ - For a 4X4 case we need to know the list [α1,α2,α3,α4]

```python
# Generate a list of n alpha values so that V iis defined
vlist=[]
N=input('Enter the number of independent alpha elements that will generate the V-M matrix')
m=[]
alpha=[]
for i in range(int(N)):
    m.append(input(''))

for item in m :
    alpha.append(float(item))
```

```
Enter the number of independent alpha elements that will generate the V-M matrix4
1
-2
3
5
```

```python
alpha
```

```
[1.0, -2.0, 3.0, 5.0]
```

```python
# A function that creates a row of the matrix
def createVMrow(alpha,colelements):
    start=1
    ratio=alpha
    progression = [start * ratio**i for i in range(colelements)]
    return(progression)
```

```python
createVMrow(2,4)
```

```
[1, 2, 4, 8]
```

```python
# A function that stacks the row and creates the matrix (list of lists)

def createVMmatrix(alpha,rowelements,colelements):
    row=[]
    for i in range(colelements):
        print(i)
        row.append(createVMrow(alpha,colelements))
```

```python
s=createVMmatrix(2,4,4)
```

```
0
1
2
3
```

```python
print(s)
```

```
None
```

```
def vm(alpha,rows,cols):
    m=[]
    for i in range(rows):
        m.append(createVMrow(alpha[i],cols))
    return m
```

```
alpha=[1,2,3,4,5]
rows=5
cols=3
s=vm(alpha,rows,cols)
[print(*j) for j in s]
# for j in s:
#     print(*j)
```

```
1 1 1
1 2 4
1 3 9
1 4 16
1 5 25
```