

# \_\_INIT\_\_.PY MODULE

```
print("enter your choice number of you want to mathematical operation you want to perform
      1.basic arithmetic operations
      2. trigonometric operations
      3.hyperbolic functions
      4.logarithms and exponential functions
      5.statistic functions
      or enter 0 to exit the loop")
p=True
while p==True:
    number=int(input("choice please."))
    if number==0:
        p=False
    elif number==1:
        import basic.py
    elif number ==2:
        import trigonometry.py
    elif number==3:
        import hyperbola.py
    elif number==4:
        import log.py
    elif number==5:
        import statistics.py
    else:
        print("enter the valid number")
```

The screenshot shows a Mac OS X desktop environment with two windows open. On the left is the IDLE Python editor window, titled '\_init\_.py - /Users/nithin/Desktop/final/\_init\_.py (3.8.5)'. It contains Python code for a menu-based calculator. On the right is the 'Python 3.8.5 Shell' window, titled 'Python 3.8.5 (v3.8.5:580fb0@18f, Jul 20 2020, 12:11:27) [Clang 6.0 (clang-600.0.57)] on darwin'. The shell window displays the code from the IDLE window and shows the user prompt 'choice plz:' followed by a list of mathematical operations. The desktop background features a dark, abstract image, and the Dock at the bottom contains various application icons.

```
print("""enter your choice number of you want to mathematical operation you want
1.basic arithmetic operations
2.trigonometric operations
3.hyperbolic functions
4.logarithms and exponential functions
5.statistic functions
or enter 0 to exit the loop""")
p=True
while p==True:
    number=int(input("choice plz:"))
    if number==0:
        p=False
    elif number==1:
        import basic.py
    elif number ==2:
        import trigonometry.py
    elif number==3:
        import hyperbola.py
    elif number==4:
        import log.py
    elif number==5:
        import statistics.py

else:
    print("enter the valid number")
```

```
Python 3.8.5 (v3.8.5:580fb0@18f, Jul 20 2020, 12:11:27)
[Clang 6.0 (clang-600.0.57)] on darwin
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: /Users/nithin/Desktop/final/final/_init_.py =====
enter your choice number of you want to mathematical operation you want to perform
choice plz:
```

# BASICS MODULE

```
import numpy as np
print("you're in the basic arithmetic operations package")
print("enter the choice please
      1.basic arithmetic operations
      2.scalar operations")
choice=int(input("enter the choice:"))
if choice==1:
    count=0
    x=2
    num=int(input("enter the number of elements you need in the array: "))
    for k in range(x):
        a=[]
        if num>0:
            for i in range(num):
                n=int(input("enter the element of the array:"))
                a.append(n)
                print(a)
            count+=1
            if count==1:
                arr1= np.array(a)
                print("arr1=",arr1)
            else:
                arr2= np.array(a)
                print("arr2=",arr2)
        else:
            print("valid number please")
# creating the class and defining the functions

class simple_math:
    def add(n,u):
        n=arr1
        u=arr2
        value= np.add( arr1, arr2)
        print("the value of adding 2 arrays",n,u,"is ", value)
    def subtract(n,u):
        n=arr1
        u=arr2
        value= np.subtract( arr1, arr2)
        print("the difference between 2 arrays",n,u,"is ", value)
    def multiply(n,u):
        n=arr1
        u=arr2
        value= np.multiply(arr1,arr2)
```

```

print("the value of multiplying 2 arrays",n,u,"is ", value)
def divide(n,u):
    n=arr1
    u=arr2
    value= np.divide(arr1,arr2)
    print("the value of dividing 2 arrays",n,u,"is ", value)
def mod(n,u):
    n=arr1
    u=arr2
    value= np.mod(arr1,arr2)
    print("the remainders of diving 2 arrays",n,u,"is ", value)
def divmod(n,u):
    n=arr1
    u=arr2
    value= np.divmod(arr1,arr2)
    print("the remainder and quotient of 2 arrays",n,u,"is ", value)
def power(n,u):
    n=arr1
    u=arr2
    value= np.power( arr1, arr2)
    print("the power 2 arrays",n,u,"is ", value)
def positive(n):
    for i in range(2):
        if i==0:
            n=arr1
            value1= np.positive(arr1)
            print("the positive values of 1st array", arr1,"is ", value1)
        else:
            n=arr2
            value2= np.positive(arr2)
            print("the positive values of 2nd array", arr2,"is ",value2)
def negative(n):
    for i in range(2):
        if i==0:
            n=arr1
            value1= np.negative(arr1)
            print("the negative values of 1st array", arr1,"is ", value1)
        else:
            n=arr2
            value2= np.negative(arr2)
            print("the negative values of 2nd array", arr2,"is ", value1)

print("""enter your choice you want to
1.addition
2.subtraction
3.multiplication
4.division

```

```

5.remainder
6.both remainder and quotient
7.power
8.positive
9.negative
or enter 0 to exit the loop""")
p=True
while p==True:
    number=int(input("choice please:"))
    if number==0:
        p=False
    elif number==1:
        s1=simple_math.add(arr1,arr2)
    elif number ==2:
        s1=simple_math.subtract(arr1,arr2)
    elif number==3:
        s1=simple_math.multiply(arr1,arr2)
    elif number==4:
        s1=simple_math.divide(arr1,arr2)
    elif number==5:
        s1=simple_math.mod(arr1,arr2)
    elif number==6:
        s1= simple_mat.divmod (arr1,arr2)
    elif number==7:
        s1=simple_math.power(arr1,arr2)
    elif number==8:
        t=int(input("enter 1 or 0 to exit"))
        if t==1:
            s1=simple_math.positive(arr1)
        else:
            s1=simple_math.positive(arr2)
    elif number==9:
        if t==1:
            s1=simple_math.negative(arr1)
        else:
            s1=simple_math.negative(arr2)
    else:
        print("enter the valid number")

elif choice==2:
    R = int(input("Enter the number of rows:"))
    C = int(input("Enter the number of columns:"))
    count=0
    x=2
    for k in range(x):

```

```

matrix =[]
if count==0:
    print("Enter the entries row wise:")
    for i in range(R):
        a =[]
        for j in range(C):
            a.append(int(input()))
        matrix.append(a)
    for i in range(R):
        for j in range(C):
            print(matrix[i][j], end = " ")
        print()
    m1=matrix
    count+=1
else:
    print("Enter the entries row wise:")
    for i in range(R):
        a =[]
        for j in range(C):
            a.append(int(input()))
        matrix.append(a)
    for i in range(R):
        for j in range(C):
            print(matrix[i][j], end = " ")
        print()
    m2=matrix
print("matrix=",m1)
print("matrix1=",m2)

class scalar:

    def mult(n):
        for i in range(2):
            n=int(input("enter the number"))
            if i==0:
                value1=m1*n
                print("the multiplied values of matrix with",m1,"are", value1)
            else:
                value2=m2*n
                print("the multiplied values of matrix with ",m2,"is ",value2)

def mat_mul(n,u):
    n=m1
    u=m2
    value=n*u
    print(" the value of multiplied matrix is",value)

```

```

def dot(n,u):
    n=m1
    u=m2
    value = np.dot(m1,m2)
    print("the dot product value of ",n,u," is ",value)
def cross(n,u):
    n=m1
    u=m2
    value = np.cross(m1,m2)
    print("the cross product value of ",n,u," is ",value)

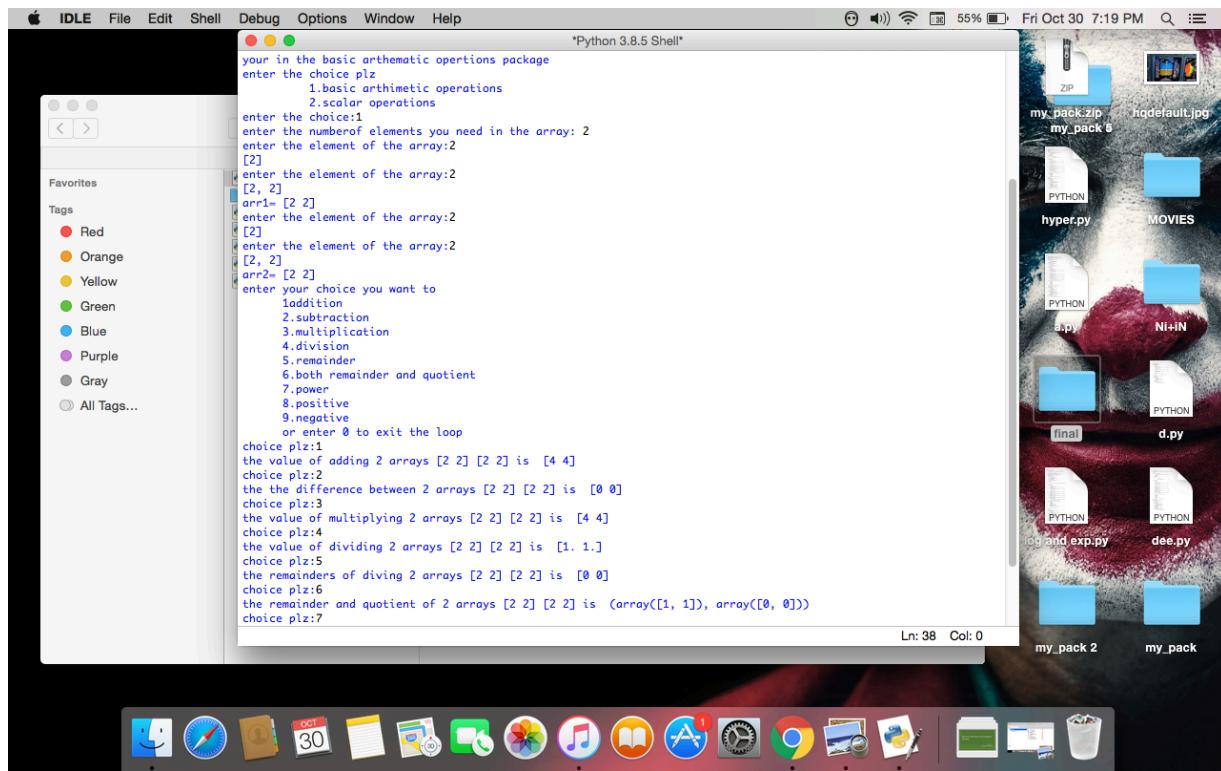
print("""enter your choice you want to
1. scalar multiplication
2. matrix multiplication
3. dot product
4. cross product
or enter 0 to exit the loop""")
p=True
while p==True:
    number=int(input("choice please:"))
    if number==0:
        p=False

    elif number ==1:
        t=int(input("enter 1 or 0 to exit"))
        if t==1:
            s1= scalar.mult(m1)
        else:
            s1= scalar.mult(m2)

    elif number==2:
        s1= scalar.mat_mul(m1,m2)
    elif number==3:
        s1=scalar.dot(m1,m2)
    elif number==4:
        s1=scalar.cross(m1,m2)

else:
    print("valid number please")

```



your in the basic arthemtic oerpations package  
enter the choice plz  
    1.basic arithmetic operations  
    2.scalar operations  
enter the choice:2  
Enter the number of rows:2  
Enter the number of columns:2  
Enter the entries rowwise:  
2  
2  
2  
2  
2 2  
2 2  
Enter the entries rowwise:  
2  
2  
2  
2  
2 2  
2 2  
matrix= [[2, 2], [2, 2]]  
matrix1= [[2, 2], [2, 2]]  
enter your choice you want to  
    1.scalar multiplication  
    2. dot product  
    3. cross product  
    or enter 0 to exit the loop  
choice plz:1  
enter 1 or 0 to exit1  
enter the number1  
the multiplied values of matrix with [[2, 2], [2, 2]] are [[2, 2], [2, 2]]  
enter the number1  
the multiplied values of matrix with [[2, 2], [2, 2]] is [[2, 2], [2, 2]]  
choice plz:2  
the .dot product value of [[2, 2], [2, 2]] [[2, 2], [2, 2]] is [[8 8]  
[8 8]]  
choice plz:3  
the cross product value of [[2, 2], [2, 2]] [[2, 2], [2, 2]] is [[0 0]  
[0 0]]

```
elif choice==2:  
    R = int(input("Enter the number of rows:"))  
    C = int(input("Enter the number of columns:"))  
    count=0  
    x=2  
    for k in range(x):  
        matrix =[]  
        if count==0:  
            print("Enter the entries rowwise:")  
            for i in range(R):  
                a =[]  
                for j in range(C):  
                    a.append(int(input()))  
                matrix.append(a)  
            for i in range(R):  
                for j in range(C):  
                    print(matrix[i][j], end = " ")  
            print()  
            m1=matrix  
        count+=1  
    else:  
        print("Enter the entries rowwise:")  
        for i in range(R):  
            a =[]  
            for j in range(C):  
                a.append(int(input()))  
            matrix.append(a)  
        for i in range(R):  
            for j in range(C):  
                print(matrix[i][j], end = " ")  
            print()  
        m2=matrix  
    print("matrix=",m1)  
    print("matrix1=",m2)  
  
class scalar:  
  
    def mult(n):  
        for i in range(2):
```



# HYPERBOLA MODULE

```
import numpy as np
print("your in the hyperbola module")
R = int(input("Enter the number of rows:"))
C = int(input("Enter the number of columns:"))
matrix = []
print("Enter the entries row wise:")
for i in range(R):
    a = []
    for j in range(C):
        a.append(int(input()))
    matrix.append(a)
arr=matrix
#creating hyperbola class

class hyperbola:
    def sinh(n):
        n=arr
        value=np. sinh(n)
        print("the value of sin(",n,") is ", value)
    def cosh(n):
        value=np. cosh(n)
        print("the value of cos(",n,") is ", value)

    def tanh(n):
        value=np. tanh(n)
        print("the value of tan(",n,") is ", value)
    def arcsinh(n):
        value=np. arcsinh(n)
        print(" the value of arcsinh(",n,") is ",value)
    def arccosh(n):
        value =np. arccosh(n)
        print("the value of arccosh(",n,") is ",value)
    def arctanh(n):
        value =np. arctanh(n)
        print("the value of arctanh(",n,") is",value)

print("""enter your choice you want to
1.sin fun
2.cos fun
3.tan fun
4. arcsinh
5. arccosh
6. arctanh
or enter 0 to exit the loop""")
```

```
p=True
while p==True:
    number=int(input("choice please:"))
    if number==0:
        p=False
    elif number==1:
        s1= hyperbola.sinh(arr)
    elif number ==2:
        s1=hyperbola.cosh(arr)
    elif number==3:
        s1= hyperbola.tanh(arr)
    elif number==4:
        s1= hyperbola.arcsinh(arr)
    elif number==5:
        s1= hyperbola.arccosh(arr)
    elif number==6:
        s1= hyperbola.arctanh(arr)
    else:
        print("enter the valid number")
```

The screenshot shows a Mac OS X desktop with a Python terminal window open in the foreground and a file browser window in the background.

**Terminal Window (Python 3.8.5 Shell):**

```
hyperbola.py - /Users/nithin/Desktop/final/final/hyperbola.py (3.8) Python 3.8.5 Shell
import numpy as np
print("your in the hyperbola module")
R = int(input("Enter the number of rows:"))
C = int(input("Enter the number of columns:"))
matrix = []
print("Enter the entries rowwise:")
for i in range(R):
    a = []
    for j in range(C):
        a.append(int(input()))
    matrix.append(a)
arr=matrix
#creating hyperbola class

class hyperbola:
    def sinh(n):
        n=arr
        value=np.sinh(n)
        print("the value of sinh(",n,") is ", value)
    def cosh(n):
        value=np.cosh(n)
        print("the value of cosh(",n,") is ", value)

    def tanh(n):
        value=np.tanh(n)
        print("the value of tanh(",n,") is ", value)
    def arcsinh(n):
        value=np.arcsinh(n)
        print("the value of arcsinh(",n,") is ",value)
    def arccosh(n):
        value =np.arccosh(n)
        print("the value of arccosh(",n,") is ",value)
    def arctanh(n):
        value =np.arctanh(n)
        print("the value of arctanh(",n,") is ",value)

print("enter your choice you want to
1.sin fun
2.cos fun
3.tan fun
4.arcsinh
5.arccosh
6.arctanh
or enter 0 to exit the loop
choice plz:1
the value of sin( [[1, 90]] ) is  [[1.17520119e+00 6.10201647e+38]]
choice plz:2
the value of cos( [[1, 90]] ) is  [[1.543088063e+00 6.10201647e+38]]
choice plz:3
the value of tan( [[1, 90]] ) is  [[0.76159416 1.          ]]
choice plz:4
the value of arcsinh( [[1, 90]] ) is  [[0.88137359 5.19298711]]
choice plz:5
the value of arccosh( [[1, 90]] ) is  [[0.          5.19292599]]
choice plz:6
enter the valid number
choice plz:0
>>> |
```

**File Browser:**

- my\_pack.zip
- my\_pack.s
- hyper.py
- Movies
- Ni+N
- final
- d.py
- dg\_and\_ex.py
- dee.py
- my\_pack 2
- my\_pack



```

import numpy as np
print("your in the logarithm and exponential module")

# creating an matrix to perform the operations

print("create your matrix")
R = int(input("Enter the number of rows:"))
C = int(input("Enter the number of columns:"))
matrix = []
print("Enter the entries row wise:")
for i in range(R):
    a = []
    for j in range(C):
        a.append(int(input()))
    matrix.append(a)
for i in range(R):
    for j in range(C):
        print(matrix[i][j], end = " ")
    print()
arr=matrix

```

# creating the class for logarithm and exponential

```

class log_exp:
    def log(n):
        n=arr
        value=np.log(n)
        print("the value of log",n,"is ", value)
    def log10(n):
        value=np.log10(n)
        print("the value of log10 of ",n," is ", value)

    def log2(n):
        value= np.log2(n)
        print("the value of log2 of ",n,"is ", value)
    def exp(n):
        value= np.exp(n)
        print(" the exponent values of ",n,"are ",value)
    def exp2(n):
        value = np.exp2(n)
        print("the exponent2 values of " ,n,"are ",value)
    def sqrt(n):
        value =np.sqrt(n)
        print("the square root values of",n,"are",value)
    def log1p(n):
        value= np.log1p(n)

```

```
print("the values of logp1 of",n,"are",value)

#printing the list of operations

print("""enter your choice you want to
1.log fun
2.log10 fun
3.log2 fun
4.exponential values
5.exponent of 2
6.square root values
7.log1p
or enter 0 to exit the loop""")

# creating the objects
```

```
p=True
while p==True:
    number=int(input("choice please:"))
    if number==0:
        p=False
        print("you're out of the loop")
    elif number==1:
        s1= log_exp.log(arr)
    elif number ==2:
        s1= log_exp.log10(arr)
    elif number==3:
        s1= log_exp.log2(arr)
    elif number==4:
        s1= log_exp.exp(arr)
    elif number==5:
        s1= log_exp.exp2(arr)
    elif number==6:
        s1= log_exp.sqrt(arr)
    elif number==7:
        s1= log_exp.log1p(arr)

    else:
        print("enter the valid number")
```

IDE File Edit Shell Debug Options Window Help

log.py - /Users/nithin/Desktop/final/final/log.py (3.8.5)

```
Python 3.8.5 (v3.8.5:580fbdb018f, Jul 20 2020, 12:11:27)
[Clang 6.0 (clang-600.0.57) on darwin
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: /Users/nithin/Desktop/final/final/log.py =====
your in the logarithm and exponential module
create your matrix
Enter the number of rows:1
Enter the number of columns:2
Enter the entries rowwise:
2
3
enter your choice you want to
1.log fun
2.log10 fun
3.log2 fun
4.exponential values
5.exponent of 2
6.square root values
7.log1p
or enter 0 to exit the loop
choice plz:1
the value of log [[2, 3]] is [[0.69314718 1.09861229]]
choice plz:2
the value of log10 of [[2, 3]] is [[0.30103 0.47712125]]
choice plz:3
the value of log2 of [[2, 3]] is [[1. 1.5849625]]
choice plz:4
the exponent values of [[2, 3]] are [[ 7.3890561 20.08553692]]
choice plz:5
the exponent2 values of [[2, 3]] are [[4. 8.]]
choice plz:6
the square root values of [[2, 3]] are [[1.41421356 1.73205081]]
choice plz:7
the values of log1p of [[2, 3]] are [[1.09861229 1.38629436]]
choice plz:0
your out of the loop
>>>
```

Ln: 8 Ln: 39 Col: 4

NiIN my\_pack 2 my\_pack

The screenshot shows a Mac desktop environment. In the foreground, there is a Python script named 'log.py' open in the IDLE editor. The code defines a class 'log\_exp' with methods for log, log10, log2, exp, and exp2. It also includes a main loop for user input. The output window of IDLE shows the execution of the script and its results. In the background, a file browser window is open, displaying a folder structure with various Python files like 'hyper.py', 'd.py', 'and exp.py', and 'dee.py'. The desktop also features a dark-themed dock at the bottom with icons for Finder, Safari, Mail, Calendar, Notes, Reminders, Stocks, iBooks, iTunes, App Store, System Preferences, Google Chrome, and a few others.

```

import numpy as np
print("you're in the statistics module")

R = int(input("Enter the number of rows:"))
C = int(input("Enter the number of columns:"))
matrix = []
print("Enter the entries rowwise:")
for i in range(R):
    a = []
    for j in range(C):
        a.append(int(input()))
    matrix.append(a)
for i in range(R):
    for j in range(C):
        print(matrix[i][j], end = " ")
    print()
arr=matrix

class statistics:
    def mean(n,axis):
        n=arr
        if axis==99:
            value=np.mean(n)
            print("the mean values of",n,"are",value)
        else:
            value=np.mean(n,axis)
            print("the mean values of",n,"are",value)
    def average(n,axis):
        n=arr
        if axis==99:
            value=np.average(n)
            print("the avarage values of ",n," are", value)
        else:
            value=np.average(n,axis)
            print("the average values of",n,"are",value)
    def median(n,axis):
        n=arr
        if axis==99:
            value=np.median(n)
            print("the median values of ",n," are ", value)
        else:
            value=np.median(n,axis)
            print("the median values of",n,"are",value)

    def variable(n):
        n=arr
        value=np.var(n)

```

```

    print("the values of variable",n,"are ", value)
def amin(n,axis):
    n=arr
    if axis==99:
        value=np.amin(n)
        print(" the minimum value of",n, " are",value)
    else:
        value=np.amin(n,axis)
        print("the minimum values of",n,"are",value)
def amax(n,axis):
    n=arr
    if axis==99:
        value =np.amax(n)
        print("the maximum values of " ,n,") is ",value)
    else:
        value=np.amax(n,axis)
        print("the maximun values of",n,"are",value)
def ptp(n,axis):
    n=arr
    if axis==99:
        value =np.ptp(n)
        print("the ptp values of",n,"are",value)
    else:
        value=np.ptp(n,axis)
        print("the ptp values of",n,"are",value)
def std(n):
    n=arr
    value=np.std(n)
    print("the standard deviation values of",n,"are",value)
def percentile(n,q,axis):
    n=arr
    if axis==99:
        value=np.percentile(a,q)
        print("the percentile of",n,"are",value)
    else:
        value=np.percentile(a,q,axis)
        print("the percentile of",n,"are",value)

# creating the objects
print("enter your choice you want to
1.mean
2.average
3.median
4.varaible
5.minimum
6.maximum
7.ptp

```

```

8.standard deviation
9.percentile
or enter 0 to exit the loop""")  

p=True  

while p==True:  

    number=int(input("choice plz:"))  

    if number==0:  

        p=False  

    elif number==1:  

        axis=int(input("enter the axis or none"))  

        if axis <=R:  

            s1=statistics.mean(arr,axis)  

        else:  

            print("you're axis is beyond the rows")  

    elif number ==2:  

        axis=int(input("enter the axis or none"))  

        if axis <=R:  

            s1=statistics.average(arr,axis)  

        else:  

            print("you're axis is beyond the rows")  

    elif number==3:  

        axis=int(input("enter the axis or none"))  

        if axis <=R:  

            s1=statistics.median(arr,axis)  

        else:  

            print("you're axis is beyond the rows")  

    elif number==4:  

        s1=statistics.variable(arr)  

    elif number==5:  

        axis=int(input("enter the axis or none"))  

        if axis <=R:  

            s1=statistics.amin(arr,axis)  

        else:  

            print("you're axis is beyond the rows")  

    elif number==6:  

        axis=int(input("enter the axis or none"))  

        if axis <=R:  

            s1=statistics.amax(arr,axis)  

        else:  

            print("you're axis is beyond the rows")  

    elif number==7:  

        axis=int(input("enter the axis or none"))  

        if axis <=R:  

            s1=statistics.ptp(arr,axis)  

        else:  

            print("you're axis is beyond the rows")  

    elif number==8:  


```

```

s1=statistics.std(arr)
elif number==9:
    q=int(input("enter the percentilt"))
    axis=int(input("enter the axis or none"))
    if axis <=R:
        s1=statistics.percentile(arr,q,axis)
    else:
        print("you're axis is beyond the rows")
else:
    print("enter the valid number")

```

The screenshot shows a Mac OS X desktop environment with two open windows:

- Python IDE (IDLE):** The left window displays the code for a script named `statistics.py`. The code defines a class `statistics` with methods for mean, average, median, minimum, maximum, ptp, standard deviation, and percentile. It also includes a main loop for user input.
- Terminal Window (Python 3.8.5 Shell):** The right window shows the execution of the script. It prompts the user for the number of rows (2), columns (2), and entries (4, 5, 7, 8). Then, it asks for a choice from a menu (1.mean, 2.average, etc.) and provides the results for each choice.

```

statistics.py - /Users/nithin/Desktop/final/final/statistics.py (3.8.5)
import numpy as np
print("your in the statistics module")

R = int(input("Enter the number of rows:"))
C = int(input("Enter the number of columns:"))
matrix = []
print("Enter the entries rowwise:")
for i in range(R):
    a = []
    for j in range(C):
        a.append(int(input()))
    matrix.append(a)
for i in range(R):
    for j in range(C):
        print(matrix[i][j], end = " ")
    print()
arr=matrix

class statistics:
    def mean(n, axis):
        if axis==99:
            value=np.mean(n)
            print("the mean values of ",n,"are",value)
        else:
            value=np.mean(n, axis)
            print("the mean values of ",n,"are",value)
    def average(n, axis):
        n=arr
        if axis==99:
            value=np.average(n)
            print("the average values of ",n," are", value)
        else:
            value=np.average(n, axis)
            print("the average values of ",n,"are",value)
    def median(n, axis):
        n=arr
        if axis==99:
            value=np.median(n)
            print("the median values of ",n," are ", value)

Ln: 10 Col: 0
my_pack 4

your in the statistics module
Enter the number of rows:2
Enter the number of columns:2
Enter the entries rowwise:
4
5
7
8
4 5
7 8
enter your choice you want to
1.mean
2.average
3.median
4.variable
5.minimum
6.maximum
7.ptp
8.standard deviation
9.percentile
or enter 0 to exit the loop
choice plz:1
enter the axis or none1
the mean values of [[4, 5], [7, 8]] are [4.5 7.5]
choice plz:2
enter the axis or none1
the average values of [[4, 5], [7, 8]] are [4.5 7.5]
choice plz:3
enter the axis or none1
the median values of [[4, 5], [7, 8]] are [4.5 7.5]
choice plz:4
enter the axis or none1
the values of variable [[4, 5], [7, 8]] are  2.5
choice plz:1
enter the axis or none1
the mean values of [[4, 5], [7, 8]] are [4.5 7.5]
choice plz:5
enter the axis or none1
the minimum values of [[4, 5], [7, 8]] are [4 7]
choice plz:6
enter the axis or none1
the maximum values of [[4, 5], [7, 8]] are [4 7]
choice plz:7
enter the axis or none1
the ptp values of [[4, 5], [7, 8]] are [3.5 1.5]
choice plz:8
enter the axis or none1
the standard deviation values of [[4, 5], [7, 8]] are [1.118 1.118]
choice plz:9
enter the axis or none1
the percentile values of [[4, 5], [7, 8]] are [50.0 50.0]
choice plz:0
exit the program

Ln: 72 Col: 4

```



# TRIGNOMETRY MODULE

```
import numpy as np
print("your in the trignometry module")
    # creating an array
R = int(input("Enter the number of rows:"))
C = int(input("Enter the number of columns:"))
matrix = []
print("Enter the entries row wise:")
for i in range(R):
    a =[]
    for j in range(C):
        a.append(int(input())))
    matrix.append(a)
arr=matrix

class trigonometric_fun:
    def sin(n):
        n=arr
        value= np.sin(n)
        print("the values of sin(",n,") is ", value)
    def cos(n):
        n=arr
        value= np.cos(n)
        print("the values of cos(",n,") is ", value)
    def tan(n):
        n=arr
        value= np.tan(n)
        print("the value of tan(",n,") is ", value)

    def arcsin(n):
        n=arr
        value= np.arcsin(n)
        print(" the values of arcsin(",n,") is ",value)
    def arccos(n):
        n=arr
        value = np.arccos(n)
        print("the values of arccos(",n,") is ",value)
    def arctan(n):
        n=arr
        value = np.arctan(n)
        print("the values of arctan(",n,") is",value)

class rad_deg:
    def radians(n):
        n=arr
```

```

value= np.radians(n)
print("the values of radians(",n,") is",value)
def degrees(n):
    n=arr
    value= np.degrees(n)
    print("the values of degrees(",n,")is",value)
def rad2deg(n):
    n=arr
    value= np.rad2deg(n)
    print("the values of rad2deg(",n,") is ",value)
def deg2rad(n):
    n=arr
    value= np.deg2rad(n)
    print("the values of deg2rad(",n,") is ",value)
def around(n,decimal):
    n=arr
    value= np.around(n,decimal)
    print("the values of around(",n,")is",value)
def floor(n):
    n=arr
    value = np.floor(n)
    print("the values of floor(",n,") is",value)
def ceil(n):
    n=arr
    value= np.ceil(n)
    print("the values of ceil(",n,") is " ,value)
print(" select the operation you want from below
    1. trigonometric functions
    2.radians and degree")
x=int(input("enter the choice"))

```

```

if x==1:
    print("enter your choice you want to
        1.sin fun
        2.cos fun
        3.tan fun
        4.arcsin
        5. arccos
        6.arctan
        or enter 0 to exit the loop")
    p=True
    while p==True:
        number=int(input("choice please:"))
        if number==0:
            p=False
            print("you're out of the loop now")

```

```

        elif number==1:
            s1= trignometric_fun.sin(arr)
        elif number ==2:
            s1= trignometric_fun.cos(arr)
        elif number==3:
            s1= trignometric_fun.tan(arr)
        elif number==4:
            s1= trignometric_fun.arcsin(arr)
        elif number==5:
            s1= trignometric_fun.acccos(arr)
        elif number==6:
            s1= trignometric_fun.arctan(arr)
        else:
            print("enter the valid number")
    if x==2:
        print(" enter the choice you need to covert
              1.to radians
              2.to degrees
              3.rad2deg
              4.deg2rad
              5.around
              6.floor
              7.ceil or enter 0 to exit")
    p=True
    while p==True:
        number=int(input("choice please:"))
        if number==0:
            p=False
            print("you're out of the loop")
        elif number==1:
            s1=rad_deg.radians(arr)
        elif number ==2:
            s1=rad_deg.degrees(arr)
        elif number==3:
            s1=rad_deg.rad2deg(arr)
        elif number==4:
            s1=rad_deg.deg2rad(arr)
        elif number==5:
            decimal=int(input("enter the decimal"))
            s1=rad_deg.around(arr,decimal)
        elif number==6:
            s1=rad_deg.floor(arr)
        elif number==7:
            s1=rad_deg.ceil(arr)
        else:
            print("enter the valid number")

```

```

else:
    print ("enter valid number please")

```

The screenshot shows a Mac OS X desktop environment. In the center is a Python IDLE window titled "trigonometry.py - /Users/nithin/Desktop/final/final/trigonometry.py (3.1)". The code in the window is as follows:

```

2.cos fun
3.tan fun
4.arcsin
5.acos
6.arctan
or enter 0 to exit the loop"""

p=True
while p==True:
    number=int(input("choice plz:"))
    if number==0:
        p=False
        print("you're out of the loop now")
    elif number==1:
        s1=trigonometric.fun.sin(arr)
    elif number ==2:
        s1=trigonometric.fun.cos(arr)
    elif number==3:
        s1=trigonometric.fun.tan(arr)
    elif number==4:
        s1=trigonometric.fun.arcsin(arr)
    elif number==5:
        s1=trigonometric.fun.acos(arr)
    elif number==6:
        s1=trigonometric.fun.arctan(arr)
    else:
        print("enter the valid number")

    if x==2:
        print(""" enter the choice you need to covert
1.to radians
2.to degrees
3.rad2deg
4.deg2rad
5.around
6.floor
7.ceil or enter 0 to exit""")

    p=True
    while p==True:
        number=int(input("choice plz:"))
        if number==0:
            p=False

```

The right side of the screen shows a file browser window titled "my\_pack 4". It displays several files and folders, including "final.zip", "MOVIES", "NI-IN", "hyperbola.py.zip", "basic.py.zip", "statistics.py.zip", and "setup.py". The desktop background features a dark, abstract image.

Apple IDLE File Edit Shell Debug Options Window Help

trigonometry.py - /Users/nithin/Desktop/final/final/trigonometry.py (3.8.5) Python 3.8.5 Shell

```

2.cos fun
3.tan fun
4.arcsin
5.acos
6.arctan
or enter 0 to exit the loop"""
p=True
while p==True:
    number=int(input("choice plz:"))
    if number==0:
        p=False
        print("you're out of the loop now")
    elif number==1:
        s1=trigonometric_fun.sin(arr)
    elif number ==2:
        s1=trigonometric_fun.cos(arr)
    elif number==3:
        s1=trigonometric_fun.arcsin(arr)
    elif number==4:
        s1=trigonometric_fun.acos(arr)
    elif number==5:
        s1=trigonometric_fun.arctan(arr)
    else:
        print("enter the valid number")
if x==2:
    print(""" enter the choice you need to covert
1.to radians
2.to degrees
3.rad2deg
4.deg2rad
5.around
6.floor
7.ceil or enter 0 to exit""")
p=True
while p==True:
    number=int(input("choice plz:"))
    if number==0:
        p=False
        choice plz:1
the values of sin( [[45, 90], [0, 1]] ) is [[0.85090352 0.89399666]
[0, 0.84147098]]
choice plz:2
the values of cos( [[45, 90], [0, 1]] ) is [[ 0.52532199 -0.44807362]
[ 1, 0.54030231]]
choice plz:3
the value of tan( [[45, 90], [0, 1]] ) is [[ 1.61977519 -1.99520041]
[ 0, 1.55740772]]
choice plz:6
the values of arctan( [[45, 90], [0, 1]] ) is [[1.54857776 1.55968567]
[0, 0.78539816]]
choice plz:5
Warning (from warnings module):

```

Ln: 81 Ln: 14 Col: 0

my\_pack.zip my\_pack.b64 default.jpg

final MOVIES

final.zip Ni+IN

hyperbola.py.zip \_init\_.py.zip

log .py.zip basic.py.zip

py.zip statistics.py.zip

my\_pack

PYTHON setup.py

