Python Fundamentals day 64

Today's agenda

- Exception handling keywords

Exception handling keywords

During handling the exceptions, we came across few keywords such as try, except, else. But there are few more keywords you should be aware of like raise and finally.

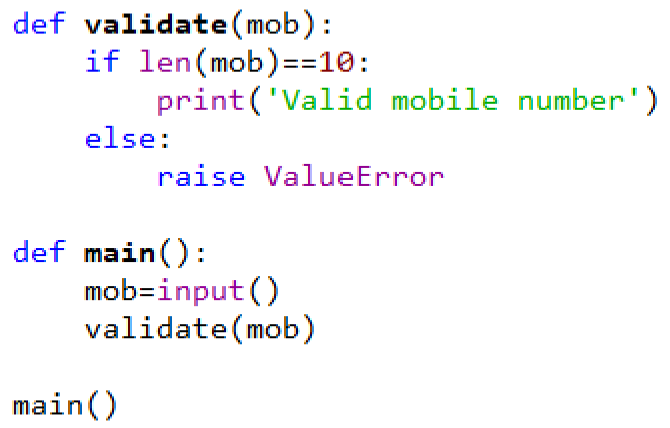
Try: The try block lets you test a block of code for errors.

Except: The except block lets you handle the error.

Else: The else block is used to define the code to be executed if no errors were raised.

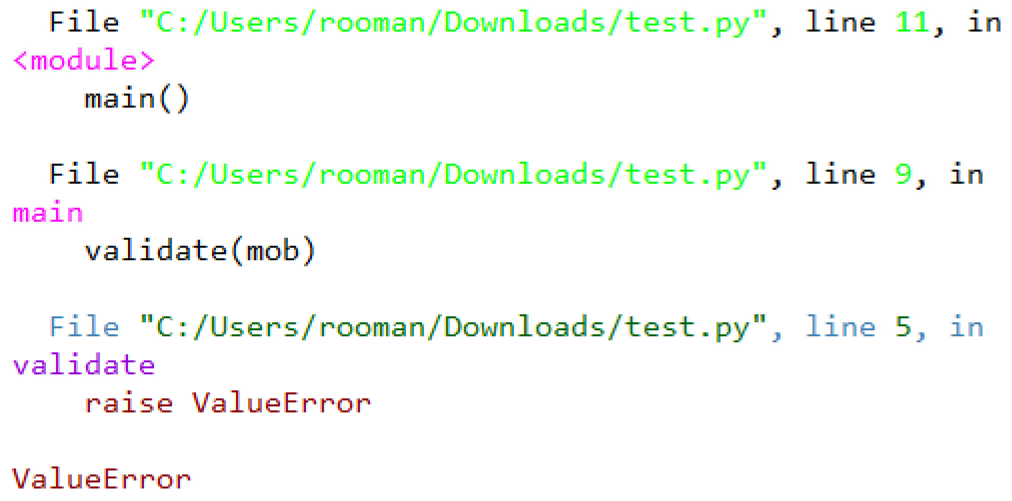
Let us see the other two keyword's functionality with the example below

Raise: As a python developer you can choose to throw an exception if the condition occurs. To throw (or raise) an exception, use the raise keyword. As shown below



```
def validate(mob):
    if len(mob)==10:
        print('Valid mobile number')
    else:
        raise ValueError

def main():
    mob=input()
    validate(mob)

main()
```

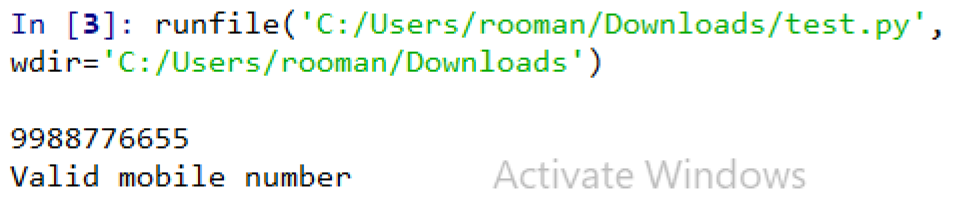
Output:



```
  File "C:/Users/rooman/Downloads/test.py", line 11, in
<module>
    main()

  File "C:/Users/rooman/Downloads/test.py", line 9, in
main
    validate(mob)

  File "C:/Users/rooman/Downloads/test.py", line 5, in
validate
    raise ValueError

ValueError
```

In the above example we are trying to just raise a value error exception.

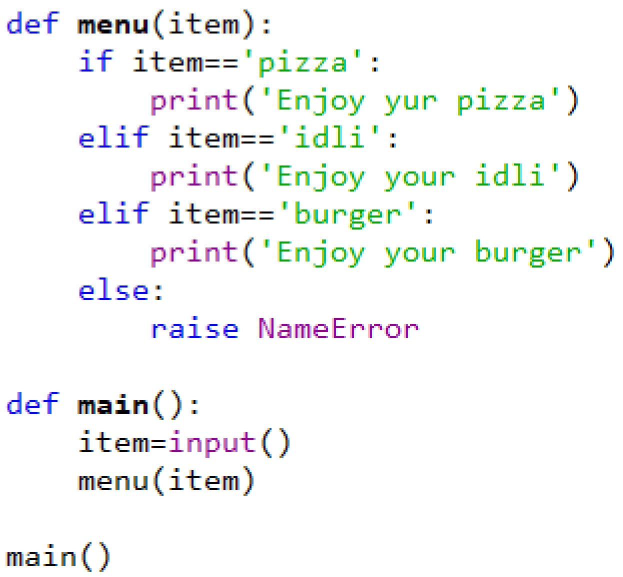
Let us see if it works fine with 10 digit input.



```
In [3]: runfile('C:/Users/rooman/Downloads/test.py',
wdir='C:/Users/rooman/Downloads')

9988776655
Valid mobile number
```

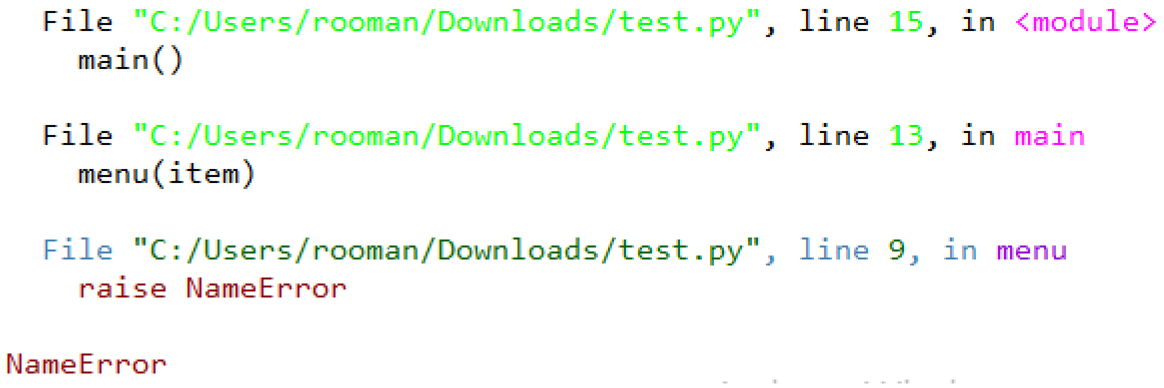
We can see the code works perfectly well with 10 digit input.

Let us take another example where we are trying to raise NameError exception



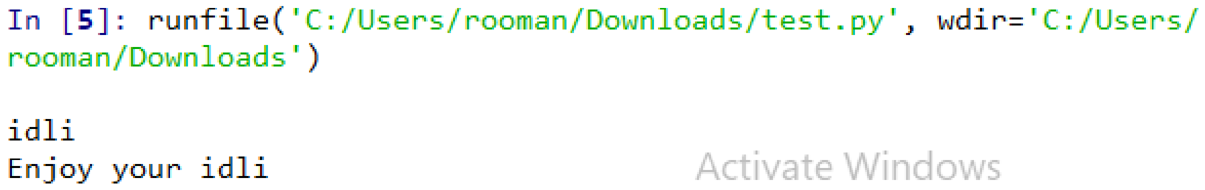
```python
def menu(item):
    if item=='pizza':
        print('Enjoy yur pizza')
    elif item=='idli':
        print('Enjoy your idli')
    elif item=='burger':
        print('Enjoy your burger')
    else:
        raise NameError

def main():
    item=input()
    menu(item)

main()
```

Output:



```
  File "C:/Users/rooman/Downloads/test.py", line 15, in <module>
    main()

  File "C:/Users/rooman/Downloads/test.py", line 13, in main
    menu(item)

  File "C:/Users/rooman/Downloads/test.py", line 9, in menu
    raise NameError

NameError
```

As the input given was pasta, and none of the condition is true else block is executed where we are trying to generate NameError exception.

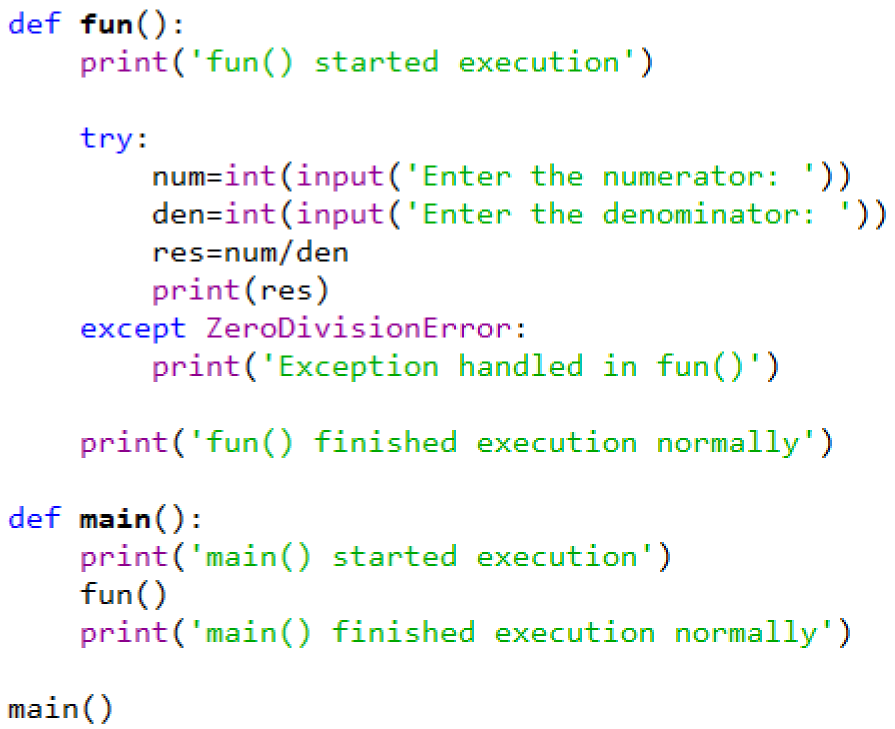


```
In [5]: runfile('C:/Users/rooman/Downloads/test.py', wdir='C:/Users/
rooman/Downloads')

idli
Enjoy your idli
```

And it works completely fine with expected food items.

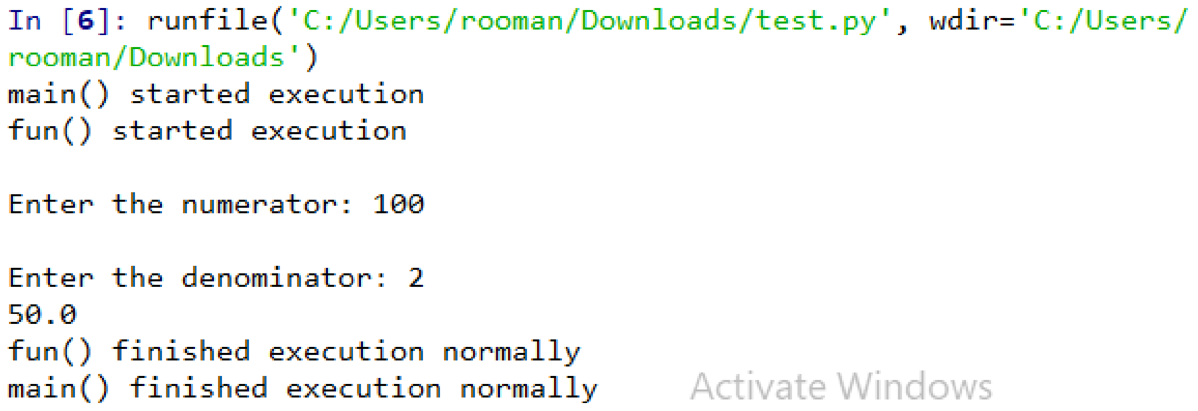
Let us next explore the finally keyword with an example

Finally: The finally block if specified will be executed regardless if the try block raises an error or not.



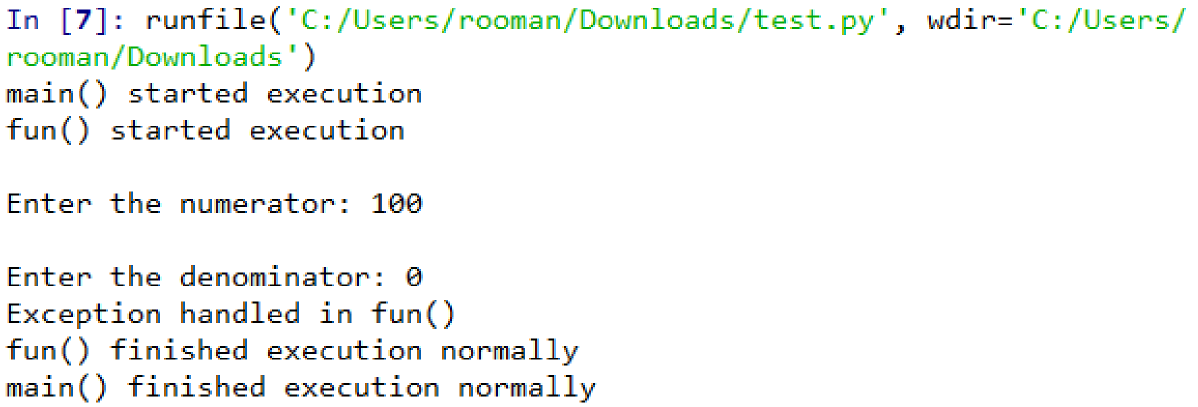
```
def fun():
    print('fun() started execution')

    try:
        num=int(input('Enter the numerator: '))
        den=int(input('Enter the denominator: '))
        res=num/den
        print(res)
    except ZeroDivisionError:
        print('Exception handled in fun()')

    print('fun() finished execution normally')

def main():
    print('main() started execution')
    fun()
    print('main() finished execution normally')

main()
```

Output:



```
In [6]: runfile('C:/Users/rooman/Downloads/test.py', wdir='C:/Users/
rooman/Downloads')
main() started execution
fun() started execution

Enter the numerator: 100

Enter the denominator: 2
50.0
fun() finished execution normally
main() finished execution normally
```

In the above output we can see the code is working fine. Let us give another incorrect input and check the working flow

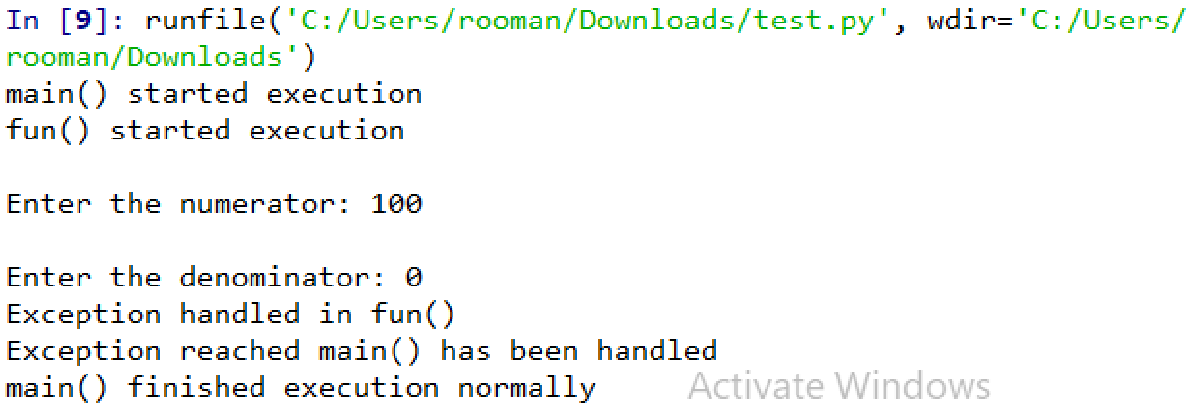


```
In [7]: runfile('C:/Users/rooman/Downloads/test.py', wdir='C:/Users/
rooman/Downloads')
main() started execution
fun() started execution

Enter the numerator: 100

Enter the denominator: 0
Exception handled in fun()
fun() finished execution normally
main() finished execution normally
```

We can notice the exception generated and handled in fun() but main() is unaware of the exception generated. Let us see below how to resolve this

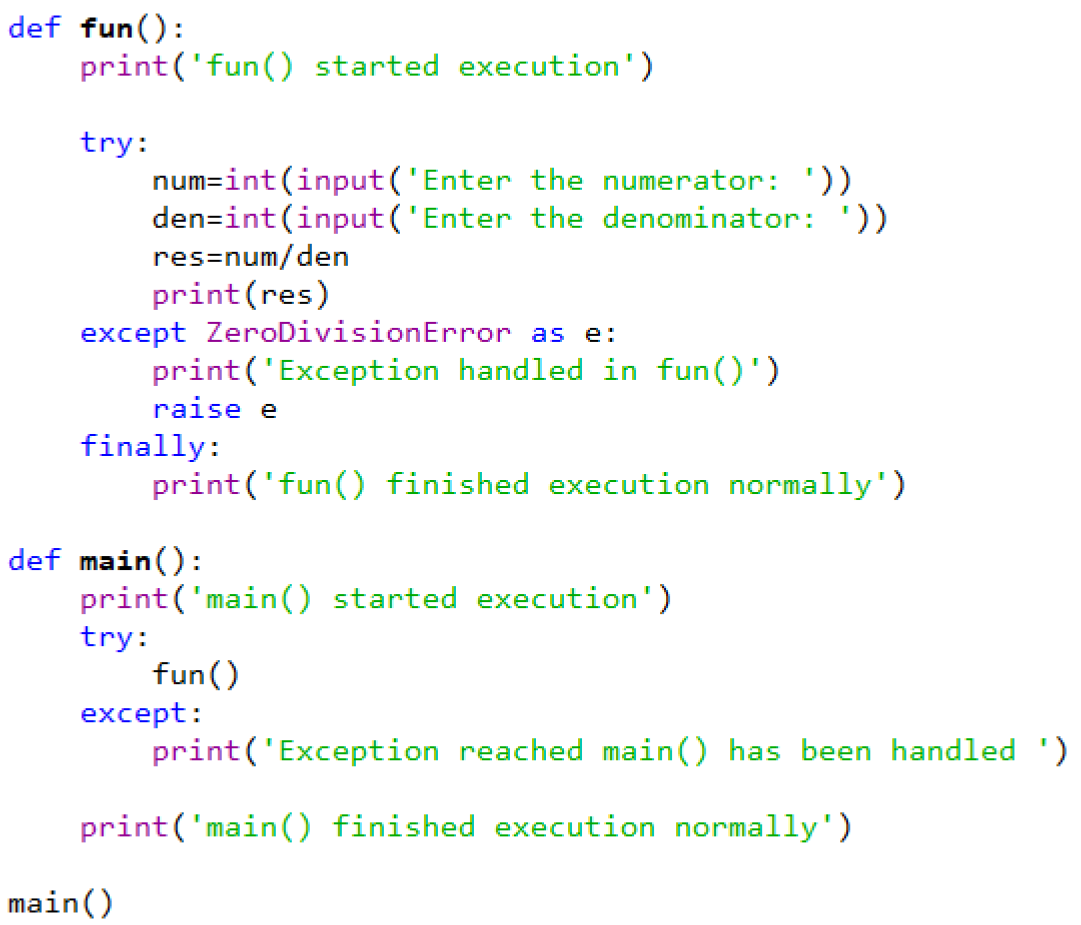
```
def fun():
    print('fun() started execution')

    try:
        num=int(input('Enter the numerator: '))
        den=int(input('Enter the denominator: '))
        res=num/den
        print(res)
    except ZeroDivisionError as e:
        print('Exception handled in fun()')
        raise e

    print('fun() finished execution normally')

def main():
    print('main() started execution')
    try:
        fun()
    except:
        print('Exception reached main() has been handled ')

    print('main() finished execution normally')

main()
```

Output:



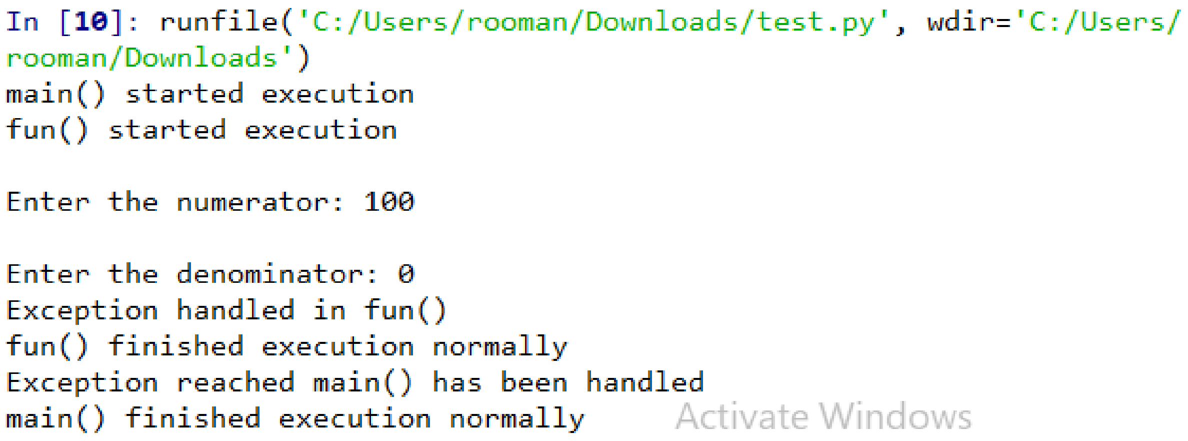
```
In [9]: runfile('C:/Users/rooman/Downloads/test.py', wdir='C:/Users/
rooman/Downloads')
main() started execution
fun() started execution

Enter the numerator: 100

Enter the denominator: 0
Exception handled in fun()
Exception reached main() has been handled
main() finished execution normally
```

In above output we can see main() has received the exception. But fun() has not completed its execution. Let’s resolve this



```python
def fun():
    print('fun() started execution')

    try:
        num=int(input('Enter the numerator: '))
        den=int(input('Enter the denominator: '))
        res=num/den
        print(res)
    except ZeroDivisionError as e:
        print('Exception handled in fun()')
        raise e
    finally:
        print('fun() finished execution normally')

def main():
    print('main() started execution')
    try:
        fun()
    except:
        print('Exception reached main() has been handled ')

    print('main() finished execution normally')

main()
```

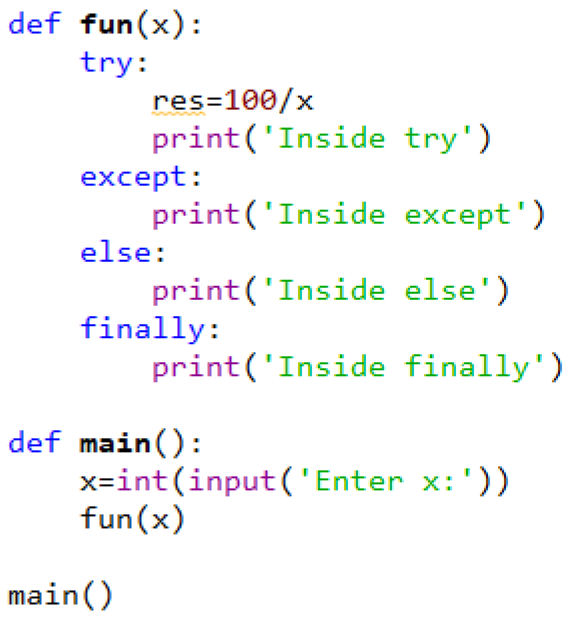
Output:



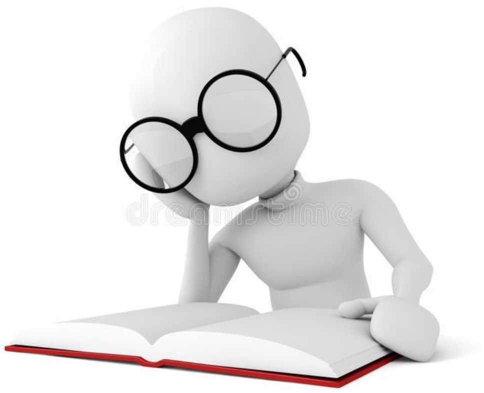
```
In [10]: runfile('C:/Users/rooman/Downloads/test.py', wdir='C:/Users/
rooman/Downloads')
main() started execution
fun() started execution

Enter the numerator: 100

Enter the denominator: 0
Exception handled in fun()
fun() finished execution normally
Exception reached main() has been handled
main() finished execution normally
```

Now all the function are executing and are completing the execution process perfectly as expected.

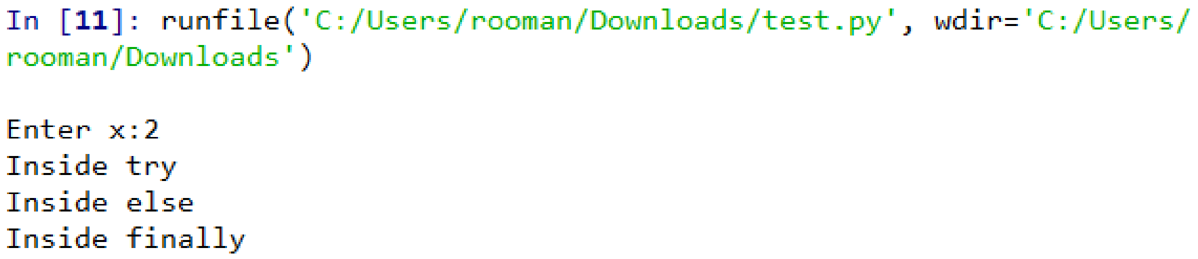
At last let us consider all the blocks and see the flow of program.



```
def fun(x):
    try:
        res=100/x
        print('Inside try')
    except:
        print('Inside except')
    else:
        print('Inside else')
    finally:
        print('Inside finally')

def main():
    x=int(input('Enter x:'))
    fun(x)

main()
```

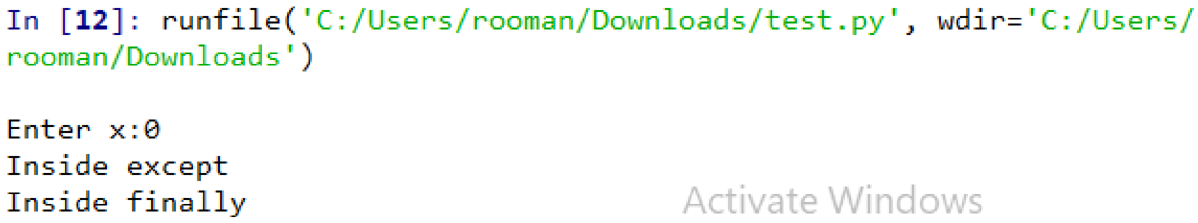


Output:



```
In [11]: runfile('C:/Users/rooman/Downloads/test.py', wdir='C:/Users/
rooman/Downloads')

Enter x:2
Inside try
Inside else
Inside finally
```

As the given input is valid, except block will not execute and as no errors/exception were encountered else block will execute and then at last irrespective of the exception generated finally block will execute.



```
In [12]: runfile('C:/Users/rooman/Downloads/test.py', wdir='C:/Users/
rooman/Downloads')

Enter x:0
Inside except
Inside finally
```

As 0 is the input, ZeroDivisionError exception object will be generated and except block is executed and then at the last finally block gets executed.