try-except

In [ ]:
```
try
except
are the keywords
```

In [ ]:
```
there is a senarion you have written 500 lines of code

python complies step by step

suppose you got the error at 250 line

the remaining lines will not execute

- eventhough if the error comes at 250 line, it should not affect the other

- error should not come

try and except
```

In [1]:
```
n1=eval(input("enter a number:"))
n2=100
add=n1+n2
print(add)
```

```
enter a number:python

---------------------------------------------------------------------------
-
NameError                                 Traceback (most recent call las
t)
Cell In[1], line 1
----> 1 n1=eval(input("enter a number:"))
      2 n2=100
      3 add=n1+n2

File <string>:1

NameError: name 'python' is not defined
```

In [ ]:
```
# entire code should implement under try block
# what ever the error comes in the code will display under except block
```

In [2]:
```
try:
    n1=eval(input("enter a number:"))
    n2=100
    add=n1+n2
    print(add)

except:
    print("error coming look at the syntax")

# when error is in code , then except block will print
```

```
enter a number:python
error coming look at the syntax
```

In [3]:
```
try:
    n1=eval(input("enter a number:"))
    n2=100
    add=n1+n2
    print(add)

except:
    print("error coming look at the syntax")
```

```
enter a number:100
200
```

In [4]:
```
try:
    n1=eval(input("enter a number:"))
    n2=100
    add=n1+n2
    print(add)

except:
    print("error coming look at the syntax")
```

```
  Cell In[4], line 2
    n1=eval(input("enter a number:"))
    ^
IndentationError: expected an indented block after 'try' statement on line
1
```

In [11]:
```
n1=eval(input("enter a number:"))
n2='100'
add=n1/n2
print(add

# name error
# value error
# syntax error
# type error
```

```
  Cell In[11], line 9
    # type error
                ^
SyntaxError: incomplete input
```

In [13]:
```
try:
    n1=eval(input("enter a number:"))
    n2=100
    add=n1+n2
    print(add)

except:
    print("error") # what is best way

# Name error
# value error
# syntax error

# what ever error comes in code , that should display
# Need to catch the error
```

```
enter a number:p
error
```

In [ ]:
```
# step-1:  we want to avoid the error
           try - except block

# step-2:  we need to catch the correct error in the excption block
           # name error
           # value eror
           # syntax error
```

In [21]:
```
try:
    n1=eval(input("enter a number:"))
    n2=0
    add=n1/n2
    print(add)

except Exception as e:
    print(e)

# as means alias name
```

```
enter a number:0
division by zero
```

```
In [ ]: try:
            number1=100
            number2=200
            add=number1+number2
            print("the addition of {} and {} is {}".format(number1,number2,add))

        except Exception as e:
            print(e)
```

```
In [ ]: try:
            number1=input("enter a number1:") # step-1: it will ask the number1='10
            number2=input("enter a number2:") # step-2: it will ask the number2='20
            add=number1+number2               # '100'+'200'='100200'
            print("the addition of {} and {} is {}".format(number1,number2,add))
        except Exception as e:
            print(e)
```

```
In [ ]: lets say i have 100 lines --
        one line has error ...
        i still want to execute the code by ignoring the line ...
        how to do it ???
```

```
In [23]: try:
             mp=eval(input("Enter Meal price:"))
             tip=eval(input("Please let me know the tip in %"))
             t_amount=(mp*tip)/100
             tot=mp+t_amount
             print("The tip amount is {} and total amount is {}".format(t_amount,tot
         except Exception as e:
             print(e)

         Enter Meal price:p
         name 'p' is not defined
```

```
In [28]: try:
             number1=input("enter a number1:") # step-1: it will ask the number1='10
             number2=input("enter a number2:") # step-2: it will ask the number2='20
             add=number1+number2 # '100'+'200'='100200'
             print("the addition of {} and {} is {}".format(number1,number2,add))

         except exception is e:
             print(e)

         enter a number1:a
         enter a number2:100
         the addition of a and 100 is a100
```

```
In [26]: 'A' + 'B'

Out[26]: 'AB'
```

```
In [29]: 'A'+'100'

Out[29]: 'A100'
```

```
In [30]: '100'+'200'

Out[30]: '100200'
```

```
In [32]: # WAP ask the user enter two numbers and add those number
         number1=100
         number2=200
         add1=number1+number2
         print(add1)


         # WAP ask the user enter numbers from keyboard and add those number
         number11=eval(input("enter number1:"))
         number22=eval(input("enter number2:"))
         add2=number11+number22
         print(add2)


         #WAP ask the user to get two random integer numbers and add those number

         300
         enter number1:600
         enter number2:700
         1300
```

```
In [39]: import random
         number1=random.randint(1,100)  # shift+tab
         print('number1 is:',number1)
         number2=random.randint(1,200)
         print('number2 is:',number2)
         add=number1+number2
         print(add)

         number1 is: 53
         number2 is: 124
         177
```

```
In [40]: help(random.randrange)

         Help on method randrange in module random:

         randrange(start, stop=None, step=1) method of random.Random instance
             Choose a random item from range(stop) or range(start, stop[, step]).

             Roughly equivalent to ``choice(range(start, stop, step))`` but
             supports arbitrarily large ranges and is optimized for common cases.
```

In [45]:
```
random.randrange(0,100,step=5)

# start=0
# stop=100
# step=5

# 0  5   10   15   20   25 30 .....

# loops clear idea


# read python everybody
```

Out[45]:
```
45
```

In [ ]:
```
import <package_name>
dir(<package_name>)
help(<package_name>.method)
```

In [35]:
```python
import random
dir(random)
```

```
Out[35]: ['BPF',
          'LOG4',
          'NV_MAGICCONST',
          'RECIP_BPF',
          'Random',
          'SG_MAGICCONST',
          'SystemRandom',
          'TWOPI',
          '_ONE',
          '_Sequence',
          '_Set',
          '__all__',
          '__builtins__',
          '__cached__',
          '__doc__',
          '__file__',
          '__loader__',
          '__name__',
          '__package__',
          '__spec__',
          '_accumulate',
          '_acos',
          '_bisect',
          '_ceil',
          '_cos',
          '_e',
          '_exp',
          '_floor',
          '_index',
          '_inst',
          '_isfinite',
          '_log',
          '_os',
          '_pi',
          '_random',
          '_repeat',
          '_sha512',
          '_sin',
          '_sqrt',
          '_test',
          '_test_generator',
          '_urandom',
          '_warn',
          'betavariate',
          'choice',
          'choices',
          'expovariate',
          'gammavariate',
          'gauss',
          'getrandbits',
          'getstate',
          'lognormvariate',
          'normalvariate',
          'paretovariate',
          'randbytes',
          'randint',
          'random',
          'randrange',
          'sample',
          'seed',
          'setstate',
          'shuffle',
          'triangular',
          'uniform',
          'vonmisesvariate',
          'weibullvariate']
```

In [36]:
```
help(random.randint)
```

```
Help on method randint in module random:

randint(a, b) method of random.Random instance
    Return random integer in range [a, b], including both end points.
```

In [ ]:
```
sir how to find in which package it is there. only by googling anaconda rep


pip freeze
```

In [ ]:
```
what is difference between randint and randrange
```

In [46]:
```
import pandas
```

In [47]:
```
dir(pandas)
```

```
 'lreshape',
 'melt',
 'merge',
 'merge_asof',
 'merge_ordered',
 'notna',
 'notnull',
 'offsets',
 'option_context',
 'options',
 'pandas',
 'period_range',
 'pivot',
 'pivot_table',
 'plotting',
 'qcut',
 'read_clipboard',
 'read_csv',
 'read_excel',
 'read_feather'
```

In [ ]:

In [ ]:

In [ ]:

In [ ]: