1.List down all the error types and check all the errors using a python program for all errors

In [14]:

```
#Not defined
a=12
print(a1)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-14-2958c0ad39b3> in <module>()
      1 #Not defined
      2 a=12
----> 3 print(a1)

NameError: name 'a1' is not defined
```

In [15]:

```
# TypeError
a="50"
b=50
print(a+b)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-15-7afeaf369e60> in <module>()
      2 a="50"
      3 b=50
----> 4 print(a+b)

TypeError: can only concatenate str (not "int") to str
```

In [16]:

```
for i range(1,10):
    print(i)
```

```
  File "<ipython-input-16-1c2bedd238d7>", line 1
    for i range(1,10):
              ^
SyntaxError: invalid syntax
```

In [17]:

```
# Paranthesis error
 print "hello"
```

```
  File "<ipython-input-17-66b8e26d5b07>", line 2
    print "hello"
                ^
SyntaxError: Missing parentheses in call to 'print'. Did you mean print("hello")?
```

In [18]:

```
# Index Error
L1=[1,2,3]
L1[3]
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
<ipython-input-18-09be1a5fde63> in <module>()
      1 # Index Error
      2 L1=[1,2,3]
----> 3 L1[3]

IndexError: list index out of range
```

In [19]:

```
#import error
import nothing
```

```
---------------------------------------------------------------------------
ModuleNotFoundError                       Traceback (most recent call last)
<ipython-input-19-532730a9649b> in <module>()
      1 #import error
----> 2 import nothing

ModuleNotFoundError: No module named 'nothing'

---------------------------------------------------------------------------
NOTE: If your import is failing due to a missing package, you can
manually install dependencies using either !pip or !apt.

To view examples of installing some common dependencies, click the
"Open Examples" button below.
---------------------------------------------------------------------------
```

In [20]:

```
#KeyError
D1={'1':"aa", '2':"bb", '3':"cc"}
D1['4']
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
<ipython-input-20-b08d41cb6197> in <module>()
      1 #KeyError
      2 D1={'1':"aa", '2':"bb", '3':"cc"}
----> 3 D1['4']

KeyError: '4'
```

In [21]:

```
#ValueError
int('xyz')
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-21-a85552ee8c9e> in <module>()
      1 #ValueError
----> 2 int('xyz')

ValueError: invalid literal for int() with base 10: 'xyz'
```

In [22]:

```
# NameError
numnber
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-22-66ecc97fbb14> in <module>()
      1 # NameError
----> 2 numnber

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

In [23]:

```
# ZeroDivisionError
x=100/0
```

```
---------------------------------------------------------------------------
ZeroDivisionError                         Traceback (most recent call last)
<ipython-input-23-de60080d639f> in <module>()
      1 # ZeroDivisionError
----> 2 x=100/0
```

```
ZeroDivisionError: division by zero
```

1. Design a simple calculator app with try and except for all use cases

In [32]:

```
def calculator():
    try:
        print('+')
        print('-')
        print('*')
        print('/')
        print('%')
        print('**')
        operation = input("Select an Operator ")
        number_1 = int(input("Enter Number 1 " ))
        number_2 = int(input("Enter Number 2 "))
        if operation == '+':
            print(number_1 + number_2)
        elif operation == '-':
            print(number_1 - number_2)
        elif operation == '*':
            print(number_1 * number_2)
        elif operation == '/':
            print(number_1 / number_2)
        elif operation == '%':
            print(number_1 % number_2)
        elif operation == '**':
            print(number_1 ** number_2)
        else:
            print('Invalid Operator')
    except Exception as x:
        print(x)
```

In [34]:

```
calculator()
```

```
+
-
*
/
%
**
Select an Operator /
Enter Number 1 10
Enter Number 2 0
division by zero
```

In [35]:

```
calculator()
```

```
+
-
*
/
%
**
Select an Operator $
Enter Number 1 10
Enter Number 2 10
Invalid Operator
```

3.print one message if the try block raises a NameError and another for other errors

In [49]:

```
try:
```

```
  print(y)
except NameError:
  print("Variable y is not defined")
except:
  print("Something else went wrong")
```

```
Variable y is not defined
```

In [50]:

```
y="Hello world"
try:
  print(y)
except NameError:
  print("Variable y is not defined")
except:
  print("Something else went wrong")
```

```
Hello world
```

4.When try-except scenario is not required

Python Exceptions are error scenarios that alter the normal execution flow of the program The process of The code inside the else block is executed if there are no exceptions raised.

5.Try getting an input inside the try catch block¶

In [54]:

```
try:
    age=int(input('Enter integer values : '))
except:
    print ('You have entered an invalid data type.')
```

```
Enter integer values : 23
```

In [55]:

```
try:
    age=int(input('Enter integer value: '))
except:
    print ('You have entered an invalid data type.')
```

```
Enter integer value: one
You have entered an invalid data type.
```