Errors :-

(i) Syntax Error -

These Syntax errors are done by coder / developer.

Ex:- and = 10

o/p:- Syntax Error: invalid Syntax

(ii) Logical Errors :-

for these logical Errors also developer is only responsible

Ex:- let, if we want to add two numbers & code has been written as below

```
a = 10
b = 20
a - b
```

o/p:- -10

It will not display any Error, but here we had done

(iii) Runtime Errors :-

for this type of Errors, users are responsible

Ex:-
```
a = int(input("enter the no"))
b = int(input("enter the no"))
print(a/b)
```

o/p:-
```
enter the no 10
enter the no 2
5.0
enter the no 10
enter the no 0
Zero Division Error
```

Try & Except can handle the above mention type Errors

Ex:-
```
a=int(input("enter the no"))
b=int(input("enter the no"))
try:
    print(a/b)
except:
    print("don't take denominator as 0")
```

Output :- enter the no 10
enter the no 0
don't take denominator as 0.

enter the no 10
enter the no f
~~dend~~ Error

Therefore,

```
a=int(input("enter no"))
try:
    b=int(input("enter no"))
    print(a/b)
except:
    print("don't take denominator as 0 & take only integers")
```

o/p try it

If we want to give statements according to condition then.

```
try:
    a = int(input("enter"))
    b= int(input("enter"))
    print(a/b)
except ZeroDivisionError:
    print("don't take don as 0").
except ValueError:
    print("don't take char/string")
```

O/p Try it yourself

→ Custom Exceptions:

```
class ValueLowException(Exception):
    pass
class ValueHighException(Exception):
    pass
num=100
try:
    n=int(input("Enter a no"))
    if n==num:
        print("your guess is correct")
    elif num<0:
        raise ValueLowException
    else:
        raise ValueHighException
except ValueLowException:
```

except ValueHighException:

print('the value is upper')

o/p: enter a no 105

the value is upper.