**Exception handeling:**

**Exception:**

* An **exception** is an event, which occurs during the execution of a program that disrupts the normal flow of the program's instructions.
* when a **Python** script encounters a situation that it cannot cope with, it raises an **exception**.
* An **exception** is a **Python** object that represents an error.
* Python provides two very important features to handle any unexpected error in your Python programs and to add debugging capabilities in them

1.exception handeling.

2.assertion

**Exception handeling**:

* When the error occurs, **Python** generate an **exception** that can be **handled**, which avoids your program to crash,this is known as exception handeling.

**Assertion:**

* It is nothing but testing our code from top to bottom and check are we follow rules or not.
* Exceptions are 2 types.

1.Inbuillt exceptions

2.userdefined exceptions

**Inbuilt exceptions**:

* The built-in exceptions can be generated by the interpreter or built-in functions.
* Except where mentioned, they have an “associated value” indicating the detailed cause of the error.
* This may be a string or a tuple containing several items of information.
* The associated value is the second argument to the [raise](https://docs.python.org/2/reference/simple_stmts.html#raise) statement.
* If the exception class is derived from the standard root class [BaseException](https://docs.python.org/2/library/exceptions.html#exceptions.BaseException), the associated value is present as the exception instance’s args attribute.
* The inbuilt exceptions are

BaseException

ArithmeticError

BufferError

LookupError

AssertionError

EOFError

AttributeError

FloatingPointError

GeneratorExit

ImportError

ModuleNotFoundError

IndexError

KeyError

NameError

OSError etc.

* Except is a keyword and except clause is used to handle our exception.
* Inorder to continue our program we must except the error so we can run remaining code.

**Syntax:**

Try:

Pass

Except:

Pass

Eg:

Try:

P(‘hello’)

Except NameError:

Print(‘there is an error’)

**o/p:**

there is an error

* We can write multiple exceptions.
* when we have to use multiple exception we use base exception class and inherit other exception in it.
* We can also use alias by using ‘as’.
* In multiple exception we must define same common error.

**Syntax:**

Except(exception 1,exception2.....) as e:

**Else:**

* It is executed when the try,except is not executing.

**Finally:**

* The must and should operation is written in finally.
* Finally executes always.

**Userdefined exceptions**:

* sometimes you may need to create custom **exceptions** that serves your purpose.
* In **Python**, **users** can **define** such **exceptions** by creating a new class..
* to raise an error use raise keyword.

**Eg:**

class Error(Exception):

class UnderAgeError(Error):

class overAgeError(Error):

class NotYetBornError(Error):

age=34

while True:

try:

inputage=int(input(‘enter age’)

if(inputage<=0):

raise NotYetBornError

elif(inputage>age):

raise OverAgeError

elif(inputage<age):

raise UnderAgeError

except overAgeError:

print(‘the person age is lessthan u are expected’)

except underAgeError:

print(‘the person age is greaterthan u are expected’)

except NotYetBornError:

print(‘the person is not borned’)

finally:

print(‘u have entered some age’)

print(‘u have successfully guessed the age of the person’)