Multiple Exceptions handeling:-

Example :-

```
Path, idx = input().split()
idx = int(idx)
File = open(Path,'r')
lst = File.read().splitlines()
Print(lst[idx])
File.close
```

Conditions of exaptions :-

1- if we Pass 1 Value only
2- Path File that doesn't exist
3- Permission error → if important File was been tried to be accessed
4- Path + string will Return error
5- - any number greater than len list Will Return error

How to hander different errors :-

```
Try:
    Path, idx = input().split()
    idx = int(idx)

    File = open(Path, 'r')        Read file content
    lst = File.Read().Splitlines
                          ↳ In
    Print(lst[idx])
except:
    ValueError:
        Print(" Value you entered is invalid")
except IndexError:
        Print(" your Index is out of bound")      → Specific exception
except FileNotFoundError:
        Print("you file is not found")
```

Good انت بتحدد ال error اللي عايز يطلع.

⇒ We can add another exception to handle other errors Not handeled.

⇒ We can group them in tuple ⇒ (exceptionerror, Index error)

Proper Resource Handling :

↳ by adding Finally at end of the code to check if the file is found or Not → if found close the file

↳ we assign it with None Before Try.

Finally:
if file is Not None:
file.close()

⇒ When we use with statment with files:

```
With open (Path, 'r') as file:
    lst = file.Readlines()
    Print (lst[Index])
```

↳ File will always be closed.

↳ all file exceptions will be handled.

This an object

⇒ except baseexception as e:
    error = str(e)
    Print (error) → you will Print the msg.