In [3]:
```
#1.Write a Python program that prompts the user to enter an integer and handles the ValueError
# exception if the user enters a non-integer value.
try:
    num=int(input("Enter a number:"))
    print("Entered number:",num)

except ValueError as e:
    print("Error: ",str(e))
```

```
Enter a number:hi
Error:   invalid literal for int() with base 10: 'hi'
```

In [4]:
```
#2.Create a program that opens a file and reads its contents. Use a try-except block to handle the FileNotFoundError
# exception and display a custom error message if the file does not exist.

try:
    file1 = open("newfile.txt","r")
    content=file1.read()
    print("File contents :",content)

except FileNotFoundError:
    print("Error!!!File does not exist")
```

```
Error!!!File does not exist
```

In [5]:
```
#3.Write a program that calculates the division of two numbers entered by the user. Use a try-except block to
# handle the ZeroDivisionError exception and display an appropriate error message if the user tries to divide by zero.

try:
    num1=int(input("Enter first number:"))
    num2=int(input("Enter second number:"))
    result=num1/num2
    print(num1,"/",num2,"=",result)

except ValueError as e:
    print("Invalid input !!!")

except ZeroDivisionError as e:
    print("Division by zero not possible!!!")
```

```
Enter first number:23
Enter second number:0
Division by zero not possible!!!
```

In [10]:

```python
#4.Create a program that attempts to connect to a website and prints the HTML content if successful.
# Use a try-except-else block to handle the requests.exceptions.RequestException exception and display an error
# message if the connection fails.

import requests

try:
    url=requests.get("htt://www.jetbrains.com/pycharm/")

except requests.exceptions.RequestException as e:
    print("Error:",str(e))

else:
    content=url.text
    print("HTML content of the webpage:",content)
```

```
Error: No connection adapters were found for 'htt://www.jetbrains.com/pycharm/'
```

In [11]:

```python
#5.Write a program that opens a file, reads its contents, and writes the contents to a new file.
# Use a try-except-finally block to ensure that the file is closed even if an exception occurs during the
# file operations.

try:
    infile=open("C:/Users/femin/PycharmProjects/pythonProject/pythonProject/BeinexPython/day17Task/fileop.txt","r",encoding="utf
    content=infile.read()
    print("File contents of fileop.txt:\n",content)


    with open("C:/Users/femin/PycharmProjects/pythonProject/pythonProject/BeinexPython/day17Task/fileop.txt", "r", encoding="utf
        with open("C:/Users/femin/PycharmProjects/pythonProject/pythonProject/BeinexPython/day17Task/newfileop.txt","w",encoding
            for line in infile:
                outfile.write(line)
    outfile=open("C:/Users/femin/PycharmProjects/pythonProject/pythonProject/BeinexPython/day17Task/newfileop.txt","r")
    print("File contents of newfileop.txt:\n",outfile.read())
    newcontent=open("New.txt","r")

except FileNotFoundError as e:
    print("Error:",e)

finally:
    if not infile.closed:
        infile.close()
    if not outfile.closed:
        outfile.close()
    print("Status of Infile closed or not :",infile.closed)
    print("Status of Outfile closed or not :",outfile.closed)
```

```
File contents of fileop.txt:
 Errors will always arise
File contents of newfileop.txt:
 Errors will always arise
Error: [Errno 2] No such file or directory: 'New.txt'
Status of Infile closed or not : True
Status of Outfile closed or not : True
```

In [12]:

```
#6.Write a Python program that reads email details (sender, recipient, subject, and body) from user
# input and sends the email using Mailtrap as the SMTP server

import smtplib
from email.mime.text import MIMEText

email_sender=input("Enter Email sender:")
email_receiver=input("Enter Email receiver:")

subject=input("Enter Subject:")
body=input("Enter body of the email:")

message=MIMEText(body)
message['From']=email_sender
message['To']=email_receiver
message['Subject']=subject

smtp_server='sandbox.smtp.mailtrap.io'
smtp_user= '52e0425359aa01'
smtp_pass='18d17d90659345'
smtp_port='2525'

server=smtplib.SMTP(smtp_server,smtp_port)
print("----Server----",server)
server.starttls()
server.login(smtp_user,smtp_pass)
server.sendmail(email_sender,email_receiver,message.as_string())
```

```
print("---------Email Sent---------")
server.quit()
```

```
Enter Email sender:feminabasheer.com
Enter Email receiver:abcgmail.com
Enter Subject:Daily Task Send mail using python
Enter body of the email:Your daily task is to send mail using python making use of smtp
----Server---- <smtplib.SMTP object at 0x000001787C877E20>
---------Email Sent---------
```

Out[12]:
```
(221, b'2.0.0 Bye')
```

In [13]:
```
#7.write a python program to send an email with multiple recipients using the smtplib library.

import smtplib
from email.mime.text import MIMEText

email_sender=input("Enter Email sender:")
list_receivers=input("Enter Email receivers seperated by space:")
email_receivers=list_receivers.split()

subject=input("Enter Subject:")
body=input("Enter body of the email:")

message=MIMEText(body)
message['From']=email_sender
message['To']=','.join(email_receivers)
message['Subject']=subject

smtp_server='sandbox.smtp.mailtrap.io'
smtp_user= '52e0425359aa01'
smtp_pass='18d17d90659345'
smtp_port='2525'

server=smtplib.SMTP(smtp_server,smtp_port)
print("----Server----",server)
server.starttls()
server.login(smtp_user,smtp_pass)
server.sendmail(email_sender,email_receivers,message.as_string())
print("---------Email Sent---------")
server.quit()
```

```
Enter Email sender:feminabasheer.com
Enter Email receivers seperated by space:femiansar.com abcbeinex.com ansiya.com
Enter Subject:Daily Task
Enter body of the email:This is a sample mail to test mail sending using python
----Server---- <smtplib.SMTP object at 0x000001787C876EF0>
---------Email Sent---------
```

Out[13]:

```
(221, b'2.0.0 Bye')
```

In [14]:

```python
#8.write a python program to handle exceptions when sending emails using Python's smtplib library.

import smtplib
from email.mime.text import MIMEText

email_sender=input("Enter Email sender:")
email_receiver=input("Enter Email receiver:")

subject=input("Enter Subject:")
body=input("Enter body of the email:")

message=MIMEText(body)
message['From']=email_sender
message['To']=email_receiver
message['Subject']=subject

smtp_server='sandbox.smtp.mailtrap.io'
smtp_user= '52e0425359aa01'
smtp_pass='18d17d90659345'
smtp_port='2525'

try:
    server=smtplib.SMTP(smtp_server,smtp_port)
    print("----Server----",server)
    server.starttls()
    server.login(smtp_usr,smtp_pass)
    server.sendmail(email_sender,email_receiver,message.as_string())
    print("---------Email Sent---------")
    server.quit()

except Exception as e:
    print("Error in sending Email:",str(e))
```

```
Enter Email sender:feminabasheer.com
Enter Email receiver:abcbeinex.com
Enter Subject:Daily Task on 12/06/2023
Enter body of the email:Please send the daily task file
----Server---- <smtplib.SMTP object at 0x000001787B25CCD0>
Error in sending Email: name 'smtp_usr' is not defined
```

In [15]:

```python
#9.Write a Python program that prompts the user to enter their age. Define a custom exception called
# InvalidAgeError that is raised when the entered age is less than 0 or greater than 150. Handle the
# InvalidAgeError exception and display an appropriate error message.

class InvalidAgeError(Exception):
    pass

try:
    age=int(input("Enter your age : "))
    if age<0 or age>150:
        raise InvalidAgeError("Invalid Age !!! Age must be between 0 to 150.")
    else:
        print("Your Age : ",age)
except InvalidAgeError as error:
    print("Error :",error)
```

```
Enter your age : 190
Error : Invalid Age !!! Age must be between 0 to 150.
```

In [16]:

```python
#10.Write a Python program that simulates a banking system. Implement a class called BankAccount with
# methods to initialize an account with a starting balance, withdraw funds, and handle a custom exception
# called NegativeBalanceError when the account balance goes below zero.

class NegativeBalanceError(Exception):
    pass

class BankAccount:
    def __init__(self, initial_balance):
        self.amount = initial_balance

    def withdraw(self, withdraw_amount):
        if self.amount - withdraw_amount > 0:
            current_amount = self.amount - withdraw_amount
            print("Current Balance :{} Rs.".format(current_amount))
        else:
            raise NegativeBalanceError("Insufficient Balance!")
try:
    initial_balance = float(input("Enter initial account balance: "))
    obj = BankAccount(initial_balance)
    withdrawal_amount = float(input("Enter withdrawal amount: "))
    obj.withdraw(withdrawal_amount)
except NegativeBalanceError as error:
    print("Error:", str(error))
```

```
Enter initial account balance: 3000
Enter withdrawal amount: 5000
Error: Insufficient Balance!
```

In [ ]: