Module

* Functions
* By its name
* Addition() -----file.py
* Functions,variables
* Manage and organize the code , reuse

Sample.py

def addition():

Print (“inside add function:”)

Def subtraction(number1,number2):

Print(number1-number2)

Example.py

Import sample

Print(sample.addition())

Test.py

From sample import subtraction

subtraction(5,2)

Math module

Min,max,

X=min(2,3) 🡺2

X=max(2,3) 🡺 3

Abs(-3.25) 🡺 3.25

Pow(2,3) 2\*2\*2

Math.sqrt(64)🡺

Math.ceil(1.4) 🡺2

Math.floor(1.4) 🡺1

Math.pi

JSON

Amazon -----payment gateway

REST API

🡺communicate with other

Amazon ---payment gateway ----

API =🡺response =🡺json

Key value

Name email

[{

“name”:”sd”,

“email”:”sds”

}

{

“name”:”sd”,

“email”:”sds”

}

{

“name”:”sd”,

“email”:”sds”

}]

Javascript object notation

=🡺 json ===work with out json data

Import json

Json -Dictionary –json

Json to python

Import json

X=’{ “name”:”xyx”,”age”:35 }’ //json

Z=json.loads(x)

Print(z[“name”])

Print(z[“age”])

Dictionary to JSON format

X={

“name”:”xyz”,

“age”:35

}

Y=json.dumps(x)

Print(y) =🡺 json format

remya@123re R \_\_E

REGEX

---example:

Import re =🡺regex

X=”The rain in spain“

y=re.search(“^The .\*spain$”,x)

if y:

print(“yes the match is correct”)

else:

print(“Not matching with the pattern”)

x=”The rain in spain”

Y=re.findall(“ai”,x)

Print(y) ==🡺

Z=re.search(“\s”,x) =🡺

Z=re.split(“\s”,x) 🡺splitted with the white space

Replace all White space with $

Re.sub(“\s”,”$”,x)

pattern = r"\d+"   # This pattern matches one or more digits  
text = "There are 123 apples"  
match = re.search(pattern, text)  
if match:  
    print("Match found:", match.group())  # Output: 123

pattern = r"\d+"  # Matches one or more digits  
text = "There are 123 apples and 456 oranges"  
matches = re.findall(pattern, text)  
print(matches)  # Output: ['123', '456']

pattern = r"Hello"  
text = "Hello, world!"  
match = re.match(pattern, text)  
if match:  
    print("Match found:", match.group())  # Output: Hello

pattern = r"\d+"  # Matches one or more digits  
text = "I have 123 apples and 456 oranges"  
new\_text = re.sub(pattern, "X", text)  
print(new\_text)  # Output: I have X a pples and X oranges

pattern = r"\s+"  # One or more whitespace characters  
text = "This is a test"  
words = re.split(pattern, text)  
print(words)  # Output: ['This', 'is', 'a', 'test']

pattern = r"(\d+)-(\d+)-(\d+)"  
text = "The event is on 2025-03-26"  
match = re.search(pattern, text)  
if match:  
    print("Year:", match.group(1))  # Output: 2025  
    print("Month-Day:", match.group(2))  # Output: 03

print("Month-Day:", match.group(3))

import re  
  
email\_pattern = r"[a-zA-Z0-9.\_%+-]

[+@[a-zA-Z0-9.-] +\.[a-zA-Z]{2,}](mailto:+@[a-zA-Z0-9.-]%20%20%20%20%20%20%20%20%20%20+\.%5ba-zA-Z%5d%7b2,%7d)" .  
text = "Please contact us at [example@example.com](mailto:example@example.com)"  
match = re.search(email\_pattern, text)  
  
if match:  
    print("Email found:", match.group())  # Output: [example@example.com](mailto:example@example.com)

Exception handling

Try

Except

Exception? Which interrupt our normal flow of our pgm

Handle it

Try ======set of code where the error occurs

Except =========handle the error

Else ===will execute when there is no error

Finally ====will execute

try:

x=10/0

except ZeroDivisionError:

print(“Divide by zer0”)

finally:

print(“Completed execution”)

try:

num=int(input(“Enter a number”)

result=10/num

except ValueError as e:

print(f“Invalid input: {e}”)

except ZeroDivisionError as e:

print(“Cannot divided by zero”)

except Exception as e:

print(“An unexcepted”)

else:

print(f”Result :{result}”)

finally:

print(“Code executed successfully”)

UsernotFound

NotEligibelforVote

custom exception:

UserNotFound

NotEligibleForRegistration

Def checkage(age):

If age<18:

Raise ValueError(“Age must be 18”)

Else

Print(“You are eligible”)

Try:

Checkage(16)

Except ValueError as e:

Print(f“Error”)

Class NotEligible(Exception):

Pass

Def checkage(age):

If age<18:

Raise NotEligible(“Age must be 18”)

Else

Print(“You are eligible”)

Try:

Checkage(16)

Except NotEligible as e:

Print(“Error”}

Try except else finally raise …create the custom exception

File handling

Test.txt

File=open(‘test.txt’,’r’)

//Content=file.read()

//Print(content)

Content1=file.readline() ==🡺

Content2=file.readlines() =====🡺

File.close()

File=open(‘test.txt’,’w’)

File.write(“hello world\n”)

File.write(“Good bye”)

File.close()

File=open(‘test.txt’,’a’)

File.write(“New content”);

File.close()

Test.txt

Existing?

Import os

If os.path.exists(“text.txt’):

With open(text.txt’,’r’) as file:

Content=file.read()

Print(content)

Else:

Print(“File does not exist!”)

Import os

Try;

With open(‘text.txt’,’r’) as file:

Data=file.read()

With open(‘text1.txt’,’w’) as filewrite:

Filewrite.write(data)

Print(“file copied successfully”)

Except FileNotFoundError:

Print(“input or ouput operation file”)

Except IOError as e:

Print(f“I/O Exception :{e})

Except Exception as e:

Print(“An unexcepted error “)

F=open(‘example.txt”,”x”) ==🡺create a new file

Import os

Os.remove(“example.txt”)

os.rmdir(“myfolder”)

^(?=.\*[a-z])

(?=.\*[A-Z])

(?=.\*\d)

(?=.\*[@$!%\*?&])[A-Za-z\d@$!%\*?&]{8,}$

Logging

-----warning

Error

info