Module==libaries in python

Import == import the libraries into code

RANDOM
```
import random

x=random.choice(["Heads", "Tails"])
print(x)
```

RANDINT
```
import random
#parameter is output boundarys
number=random.randint(1,13)
print(number)
```

SHUFFLE
```
import random
#parameter is output boundarys
number=["Car","Bird","Cat"]
random.shuffle(number)
print(f'{number}')
```

SHUFFLE USING FOR LOOP
```
import random
#parameter is output boundarys
number=["Car","Bird","Cat"]
random.shuffle(number)
for num in number:
    print(f'{num}')
```

STATS LIBARIES

AVERAGE
‘
```
import statistics
#gruop of things have to be in list
x=statistics.mean([3,4,5,6])
print(x)
```

COMMAND LINE ARGUMENTS

SYS
```
import sys
```

```
#argsv0 is program name
#some modules dont make you run file manually
print("Favorite food:", sys.argv[1])
print ("Favorite Sides:", sys.argv[2])
```

SYS.EXIT AND FOR LOOP

```
import sys

if len(sys.argv)<2:
    sys.exit("Too few arguments")
elif len(sys.argv)>4:
    sys.exit("Too many arguments")

for arg in sys.argv:
    print("The list is", arg)
```

EMOJI

```
import emoji

def main():
    text=emoji()
    if text == "Atm":
        print(emoji.emojize(':ATM_sign:'))
    elif text == "Tree":
        print(emoji.emojize(':Christmas_tree:'))
    elif text == "Mexico":
        print(emoji.emojize(':Mexico:'))
    elif text == "New":
        print(emoji.emojize(':NEW_button:'))
    else:
        print(emoji.emojize(':avocado:'))

def emoji():
    while True:
        try:
            text=input(“Input”)
            text=text,title()
        except ValueError:
            pass
        else:
            Return text
```

FIGMENT

```
import pyfiglet
import sys

if len(sys.argv)==0:
   sys.exit("Too few arguments")
elif len(sys.argv)>2:
   sys.exit("Too many arguments")

for arg in argv:
   result = pyfiglet.figlet_format(arg)
   print(result)
```

CONTROL D

```
#USE EOFERROR to activate control-d
import inflect

p=inflect.engine()
fam=[ ]
while True:
   try:
      inputi = input("Enter Name")
      inputi = inputi.title()
      fam.append(inputi)


   except EOFError:
     new = p.join(fam)
     print(new)
     break
   else:
      Continue
```

INFLECT

```
import inflect

p=inflect.engine()
fam=[]
while True:
   try:
      inp = input("Enter Name")
      inp = inp.title()
      fam.append(inp)
```

```
    except EOFError:
        p=p.join(fam)
        print(f'Adieu, Adieu{p}')
        break
    else:
        Continue
```

RANDOM GUESSING GAME

```
import random

def main():
        r=get_num()

        while True :
            i=int(input("GUESS"))
            if i < r:
                print("Too Small")
                continue
            elif i > r:
                print("Too Large")
                continue
            elif i == r:
                print("Just Right")
                break


def get_num():
    try:
        n=int(input("Enter limit"))
        if n < 2:
            n=int(input("Enter limit"))
        else:
            s=random.randint(1,n)
    except ValueError:
        pass
    else:
        return s

main()
```

SLICING

```
import sys
sys.argv=["Carter","Mike"]
if len(sys.argv)<2:
    sys.exit("Too few arguments")
```

```
#Prints each one indiviually
for arg in sys.argv[0:]:
   print("The list is", arg)
```
PACKAGe- OUtsourced library(Libaries can be acquired at pypi)

APIS-THIRD PARTY TOOLS

JSON-Stores data and can be read in any language

USING APIS to extract data and JSON to filter it

```
import json
import requests
import sys

if len(sys.argv) != 2:
	sys.exit()

response=requests.get("https://itunes.apple.com/search?entity=song&limit=7&term=" +
sys.argv[1])
o = response.json()
for result in o["results"]:
	print(result["trackName"])
```
CREATING A CUSTOM PACKAGE

#Package

```
def main():
	hello("world")
	goodbye("world")

def hello(name):
	print(f'hello, {name}')

def goodbye(name):
	print(f'Goodbye, {name}')

#makes sure main isnt always called
if __name__ == "__main__":
   main()
```
#MAIN

```
import sys
#import just main wont work
from Main import hello

if len(sys.argv)==2:
```

```
    hello(sys.argv[1])
```

UNIT TEST

SQUARE TEST

Code

```
def main():
        x=int(input("What is x?"))
        print("x squared is", square(x))


def square(n):
        return n*n


if __name__ == "__main__":
        main()
```

TEST

```
from Main import square
import pytest


def test_positive():
    assert square(2)==4
    assert square(0)==0
    assert square(4)==16

def test_negative():
    assert square(-4)==16
    assert square(-8)==64
    assert square(-10)==100

def test_zero():
    assert square(0)==0

def test_main():
    with pytest.raises(TypeError):
        square("Cat")
```

CODE

```
def main():
        x=input("What is your name")
        print(hello(x))
```

```
def hello(to="world"):
        return f'Hello {to}'
## To do test using package

if __name__ == "__main__":
        main()
```

EXAMPLE TEST

```
from Main import hello
#when testing make sure main class returns objects outerwise wont work

def test_main():
    assert hello("Jesus")=="Hello Jesus"

def test_hello():
    assert hello()=="Hello world"
```

NOTE-seperate test by categories

CODE

```
def main():
    tex = input("Input: ")
    output=shorten(tex)
    print(output)



def shorten(word):
    vowels = ['a', 'e', 'i', 'o', 'u','A','E','I','O','U']
    newText = ""

    for i in range(len(word)):
            if word[i] not in vowels:
                    newText += word[i]
    #seperate out put from loop and use spaces instead of tabs

    word = newText
    return word


if __name__ == "__main__":
    main()
```

TEST

```
from Main import shorten

def test_novowels():
    assert shorten("hyrt")=="hyrt"

def test_allvowels():
    assert shorten("aeiou")==""

def test_mix():
    assert shorten("heart")=="hrt"

from Main import is_valid
#Boolean test should be True or False not equvilant result
def test_isvalid():
    assert is_valid("we123")==True
    assert is_valid("CS50")==True
    assert is_valid("Aq1223")==True

def test_isinvalid():
    assert is_valid("6788,788")==False
    assert is_valid("AA022")==False
    assert is_valid("1233AAAAA")==False
```

INPUT OUTPUT

```
name = input("What is your name")
#creating and adding text to file
with open("names.txt", "a") as file:
    file.write(f'{name}\n')
```

READ FILE

```
with open("names.txt", "r") as file:
    for line in file:
        #rstrip take away any extra spaces
        print("hello", line.rstrip())
```

SORT FILE

```
names = []

with open("names.txt") as file:
    for line in file:
        #rstrip take away any extra spaces
        names.append(line.rstrip())

for names in sorted(names):
    print(f'Hello,{names}')
```

COMMA SEPERATED VALUES

```
#Read and separate values in CSV FILE
import csv

students=[]
#how to sort dictonaries in python3
with open("stuf.csv") as file:
    reader=csv.reader(file)
    for name, clas in reader:
        name=name.strip()
        students.append({"name":name,"class":clas})
#lambda is a function that does not have a name

for student in sorted(students, key=lambda student:student["name"]):
    print(f"{student['name']} is in class {student['class']}")
```

WRITE INTO A CSV FILE

```
import csv

name = input("What is your name:")

clas=int(input("What is their grade:"))

with open("stuf.csv", "a") as file:
    writer=csv.writer(file)
    writer.writerow([name,clas])
```