MODULES

Modules in Python are files containing Python code that define functions, classes, and variables.

They provide a way to organize and reuse code. Modules can be imported into other Python scripts to access the code and functionality they provide.

They help in achieving modularity and code separation, making code maintenance and collaboration easier.

Python provides a rich library of modules for various purposes, and you can also create your own custom modules.

FREQUENTLY USED MODULES

- OS Module
- Sys Module
- Math Module
- Time Module
- Datetime Module
- Calendar Module

Hint: Math module has already seen in chap -6 Functional Programming

OS MODULE

The OS module in Python provides functions for interacting with the operating system.

It allows you to perform various file and directory operations, such as creating, deleting, renaming, and checking the existence of files and directories. It provides functions like **os.mkdir()**, **os.remove()**, **os.rename()**, and **os.path.exists()**.

Below example shows the how to get current directory, change directory and make new directory:

```
os.chdir("./Documents")


# make new directory
newFolder = "testingOs"
os.mkdir(newFolder)


cdir = getDir()
print(cdir)
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  COMMENTS

PS C:\Users\Ram prasath>  c:; cd 'c:\Users\Ram prasath'; & 'C:\Users\Ram prasath\AppData\Local\Programs\Python\Python31
.exe' 'c:\Users\Ram prasath\.vscode\extensions\ms-python.python-2023.12.0\pythonFiles\lib\python\debugpy\adapter/../..\
launcher' '64418' '--' 'c:\Users\Ram prasath\main.py'
C:\Users\Ram prasath\Documents
PS C:\Users\Ram prasath>  c:; cd 'c:\Users\Ram prasath'; & 'C:\Users\Ram prasath\AppData\Local\Programs\Python\Python31
.exe' 'c:\Users\Ram prasath\.vscode\extensions\ms-python.python-2023.12.0\pythonFiles\lib\python\debugpy\adapter/../..\
launcher' '64463' '--' 'c:\Users\Ram prasath\main.py'
C:\Users\Ram prasath\Documents
PS C:\Users\Ram prasath> cd .\Documents\
PS C:\Users\Ram prasath\Documents> cd .\testingOs\
PS C:\Users\Ram prasath\Documents\testingOs> D
```

```
import os

#  list all the files and directories
print(os.listdir())


# to remove the specific file using remove()
dirr = os.getcwd()
path = os.path.join(dirr, "subMain.py")
os.remove(path)

# if you want to remove the directory then use rmdir()
path = os.path.join(dirr, "removeTesting")
os.rmdir(path)
```

SYS MODULE

The sys module in Python provides functionality for interacting with the system.
It allows you to access command-line arguments through sys.argv, retrieve system-specific information with functions like sys.version, and control standard input/output streams using sys.stdin and sys.stdout.

Additionally, sys.exit() can be used to explicitly exit the program with an optional exit status. Overall, the sys module offers essential features for system-related tasks in Python programming.

```python
import sys

# it shows the version of python interpreter
print(sys.version)

# ----------------------------------------------------------------

# read the input from the user
def main():
    # Read user input from stdin
    name = sys.stdin.readline().strip()

    # Display a personalized greeting
    print("name", name)

main()

#----------------------------------------------------------------

# stdout means 'STANDARD output STREAM'
sys.stdout.write("Welcome to Python")

# ----------------------------------------------------------------
def main():
    try:
        # Perform some operation that may raise an exception
        result = 10 / 0
    except Exception as e:
        # Print the error message to sys.stderr
        sys.stderr.write("An error occurred: " + str(e))

main()
```

TIME MODULE

```python
import time

print(time.gmtime(0))


timeInSec = time.time()
print("current time in seconds", timeInSec)
print("current time in proper format ", time.ctime(timeInSec))

# time.sleep() used to delay in given seconds
for i in range(10):
    time.sleep(1)
    print(i)


current_time = time.localtime()
```

```python
# strftime is used to format the local time
formatted_time = time.strftime("%Y-%m-%d %H:%M:%S", current_time)

print(formatted_time)

#  represents the time offset in seconds between the local time zone and Coordinated Universal Time (UTC).
print(time.timezone)

# Daylight is the practice of adjusting the clock forward by one hour
# during certain periods, typically in the summer, to make better use of daylight
# and extend daylight hours in the evening.
# It is used to optimize energy usage and align human activities with natural daylight patterns.
print(time.daylight)
```

DATETIME MODULE

```python
import datetime


# getting current date and time
print(datetime.datetime.now())
# ----------------------------------------------------------------

# Create a datetime object for July 1, 2023, 10:30 AM
specific_datetime = datetime.datetime(2023, 7, 1, 10, 30)
print(specific_datetime)

# ----------------------------------------------------------------

print(datetime.datetime.now().date())
print(datetime.datetime.now().time())
print(datetime.datetime.now().weekday())

# ----------------------------------------------------------------

# Returns the number of seconds since January 1, 1970, as a floating-point value.
print(datetime.datetime.now().timestamp())
```

CALENDER MODULE

```python
import calendar


# getting calender for specific dates
print(calendar.month(2017, 1))

# ----------------------------------------------------------------

# getting calender for whole year
print(calendar.calendar(2020))
```

```python
# ------------------------------------------------------------

# find the year is leap year or not
print(calendar.isleap(2023))

# Returns the day of the week (0-6, where Monday is 0) for a given date.
print(calendar.weekday(2023, 7, 7))
```

HOW TO MIGRATE PYTHON PROGRAMS AS MODULE?

Lets create a basic **'Calculator'** module step by step.

- Create a directory(folder) in your file manager called "calculator" to hold your calculator module.
- Inside the "calculator" directory, create a file called "operations.py". This file will contain the functions for performing various calculations.
  write your functions for calculations like addition, subtraction, multiplication, etc,. and save the file.
- In the "calculator" directory, create another file called "calculator.py". This file will serve as the main entry point for the calculator module.
- In "calculator.py", import the functions from "operations.py" so that they can be accessed when the calculator module is imported:

```python
from .operations import add, subtract, multiply, divide
```

- In the "calculator" directory, create an empty file called "**init**.py". This file will make the "calculator" directory a Python package.
- Your calculator module is now ready to be used. You can use it in another Python script like this:

```python
from calculator.calculator import add, subtract, multiply, divide

result = add(5, 3)
print("Addition:", result)

result = subtract(10, 4)
print("Subtraction:", result)

result = multiply(2, 6)
print("Multiplication:", result)
```

```
result = divide(10, 2)
print("Division:", result)
```

TYPES OF IMPORTING MODULES

```
import math
```

import module name as alias:

```
import numpy as np
```

```
from random import randint
```

```
from math import *
```

```
from datetime import datetime as dt
```