1. Add the current date to the text file today.txt as a string.

**Answer :**

from datetime import date

current\_date = date.today().strftime("%Y-%m-%d")

with open("today.txt", "a") as file:

file.write(current\_date + "\n")

This code imports the date class from the datetime module and gets the current date using **date.today().** Then, it formats the date as a string in the format **"YYYY-MM-DD"** using the **strftime** method.

The code then opens the file "today.txt" in append mode ("a") and writes the current date followed by a newline character ("\n") to the file. The with statement ensures that the file is properly closed after writing.

2. **Read the text file today.txt into the string today\_string.**

**Answer :**

with open("today.txt", "r") as file:

today\_string = file.read()

print(today\_string)

This code opens the **"today.txt"** file in read mode ("r") using a with statement to ensure proper file handling. It then uses the read() method to read the entire contents of the file and assigns it to the variable **today\_string**.

Finally, the code prints the value of today\_string, which will display the contents of the **"today.txt"** file.

**3. Parse the date from today\_string.**

**Answer :**

from datetime import datetime

date\_format = "%Y-%m-%d" # Format of the date in the today\_string

parsed\_date = datetime.strptime(today\_string.strip(), date\_format).date()

print(parsed\_date)

we assume that the today\_string variable contains a date string in the format **"YYYY-MM-DD",** similar to what was written in the "today.txt" file. You can adjust the date\_format variable if the actual format is different.

The code uses the strptime() method from the datetime module to parse the date string. The strip() method is called to remove any leading or trailing whitespace from the today\_string before parsing.

The result is stored in the parsed\_date variable, which will be a date object representing the parsed date.

**4. List the files in your current directory**

**Answer :**

import os

directory = "." # Specify the directory path here, "." represents the current directory

files = os.listdir(directory)

for file in files:

print(file)

**the os.listdir()** function is used to retrieve a list of files and directories within the specified directory. By providing the path of the directory you want to list, you can replace the directory variable with your desired directory path.

**5. Create a list of all of the files in your parent directory (minimum five files should be available).**

**Answer :**

import os

parent\_directory = os.path.dirname(os.getcwd())

files = os.listdir(parent\_directory)

file\_list = []

for file in files:

if os.path.isfile(os.path.join(parent\_directory, file)):

file\_list.append(file)

print(file\_list)

The os module is used to manipulate file paths and retrieve file information.

The parent\_directory variable is set as the parent directory of the current working directory **(os.getcwd()).** You can modify this if you have a different reference point in mind.

The **os.listdir()** function is used to obtain a list of files and directories within the parent directory.

A for loop iterates over the files in the parent directory. The **os.path.isfile()** function is used to check if each item is a file (as opposed to a directory), and if so, it is appended to the **file\_list list.**

**6. Use multiprocessing to create three separate processes. Make each one wait a random number of seconds between one and five, print the current time, and then exit.**

**Answer :**

import multiprocessing

import time

import random

from datetime import datetime

def worker():

wait\_time = random.randint(1, 5)

time.sleep(wait\_time)

current\_time = datetime.now().strftime("%H:%M:%S")

print(f"Process {multiprocessing.current\_process().name} waited for {wait\_time} seconds. Current time: {current\_time}")

if \_\_name\_\_ == '\_\_main\_\_':

processes = []

for i in range(3):

p = multiprocessing.Process(target=worker)

p.start()

processes.append(p)

for p in processes:

p.join()

**7. Create a date object of your day of birth.**

**Answer :**

from datetime import date

day\_of\_birth = date(2000, 1, 1) # Replace with your actual day of birth

print(day\_of\_birth)

the date class from the datetime module is imported. The date constructor is then used to create a date object representing the specified day of birth. You need to replace the arguments 2000, 1, and 1 with the year, month, and day of your actual day of birth, respectively.

Finally, the code prints the day\_of\_birth, which displays the created date object representing the day of birth.

**8. What day of the week was your day of birth?**

**Answer :**

from datetime import date

day\_of\_birth = date(2000, 1, 1) # Replace with your actual day of birth

day\_of\_week = day\_of\_birth.strftime("%A")

print(day\_of\_week)

the date class from the **datetime** module is imported. The date constructor is then used to create a date object representing the specified day of birth. You need to replace the arguments 2000, 1, and 1 with the year, month, and day of your actual day of birth, respectively.

The **strftime("%A")** method is called on the **day\_of\_birth** object with the format **specifier %A** to obtain the full weekday name for the date.

**9. When will you be (or when were you) 10,000 days old?**

**Answer :**

from datetime import date, timedelta

reference\_date = date(2000, 1, 1) # Replace with your desired reference date

future\_date = reference\_date + timedelta(days=10000)

print(future\_date)

the date class and **timedelta** class from the datetime module are imported. The date constructor is used to create a date object representing the reference date. You can replace the arguments 2000, 1, and 1 with the year, month, and day of your desired reference date, respectively.

The **timedelta(days=10000**) object is created to represent a time delta of 10,000 days. The reference date is then added to this **timedelta** using the + operator, resulting in a future date that is 10,000 days after the reference date.