**Python – Assignment**

Problem Statement – Pseudocode

START PROGRAM  
SET INSTRUCTIONS\_FILE = "instructions.txt"  
  
IF ARG = 'file-ops' or ARG ='--fo'  
 WHILE True:  
 PRINT "Options:"  
 PRINT "1. Add an instruction"  
 PRINT "2. Execute instructions"  
 PRINT "3. Exit"  
   
 INPUT choice   
   
 IF choice == "1":  
 PRINT "Enter your instruction:"  
 INPUT instruction  
 OPEN INSTRUCTIONS\_FILE IN append mode  
 GET current timestamp  
 WRITE instruction and timestamp TO file  
 CLOSE file  
 PRINT "Instruction logged!"  
   
 ELSE IF choice == "2":  
 CHECK IF INSTRUCTIONS\_FILE exists  
 IF NOT exists:  
 PRINT "No instructions found."  
 ELSE:  
 OPEN INSTRUCTIONS\_FILE IN read mode  
 READ all lines from file  
 FOR EACH line IN lines:  
 SPLIT line INTO timestamp and instruction  
 TRY:  
 IF instruction == "time":  
 DISPLAY current time  
 ELSE IF instruction == "date":  
 DISPLAY current date  
 ELSE:  
 RAISE error for unknown instruction  
 CATCH error:  
 PRINT "Error with instruction: ", instruction  
 PRINT "Error details: ", error  
 CLOSE file  
   
 ELSE IF choice == "3":  
 PRINT "Goodbye! Check 'instructions.txt' for your instructions."  
 BREAK  
   
 ELSE:  
 PRINT "Invalid choice. Please try again."  
   
 END WHILE  
  
ELIF ARG ='Pandas' or ARG ='--p':  
 WHILE True:  
 PRINT "Options:"  
 PRINT "1. Open/Read CSV"  
 PRINT "2. Exit"  
   
 INPUT choice  
   
 IF choice == "1":  
 USER INPUT = "file-name.csv"  
 IF NOT exists:  
 PRINT "No file found."  
 ELSE:  
 OPEN CSV  
 PRINT CSV-Content  
  
ELSE  
 PRINT "No Argument  
"  
 PRINT "User --fo for file operations (.Txt)"  
 PRINT "User -p for Pandas (.csv)"  
END PROGRAM

**Solution to the Problem Statement**

**Main.py Description**

**main.py** Serves as the Entry point for a Python program that supports two distinct functionalities: File Operations and Pandas CSV Operations. The script uses command-line arguments to determine which operation mode to activate. It is structured in a modular way, relying on functions defined in separate modules (file\_op and pandas\_op) for specific operations.

* **Argument Handling:**

The script checks if a command-line argument is provided.

Depending on the argument, it decides whether to activate File Operations Mode or Pandas Operations Mode.

* **Modes of Operation:**

**File Operations Mode (--fo or file-ops):**

* + Delegates tasks to the file\_operation function from the file\_op module.
  + Likely includes features like logging instructions and executing them from a file (instructions.txt).

**Pandas Operations Mode (--p or Pandas):**

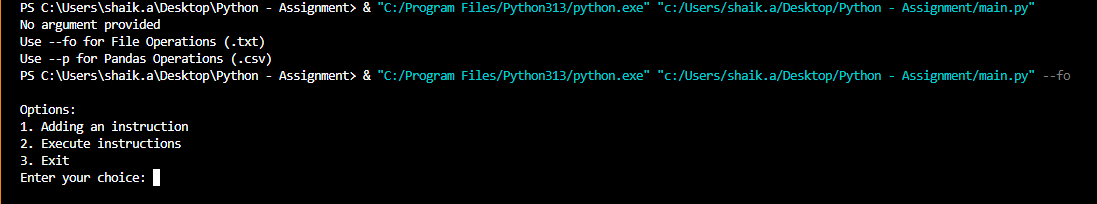
* + Delegates tasks to the pandas\_operation function from the pandas\_op module.
  + Includes features like opening and reading CSV files.

**Code For Main.py**

**A screenshot of a computer program

Description automatically generated**

**Output – Main.py**

****

**1. File Operations Mode (--fo )**

This mode is used to manage instructions stored in a text file (instructions.txt). It includes the following features:

* **Log Instructions**:  
  Users can log an instruction into the file along with the current timestamp. For example, logging "time" will record it with the date and time of entry. This helps maintain a historical record of user instructions.
* **Execute Instructions**:  
  The program reads instructions from the file and performs predefined operations:
  + "time": Displays the current system time.
  + "date": Displays the current date.  
    If the instruction is unrecognized, the program notifies the user with an error.
* **Exit**:  
  This option allows users to exit

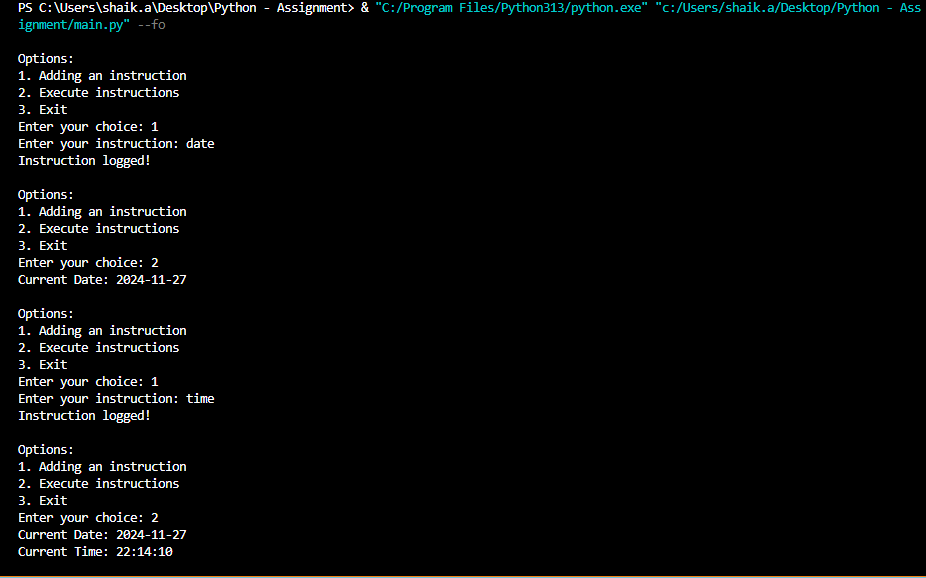
**File Operations** – Code



**Clear Image of Source Code**



File Operations – Output



**2. Pandas Operations Mode (--p or Pandas)**

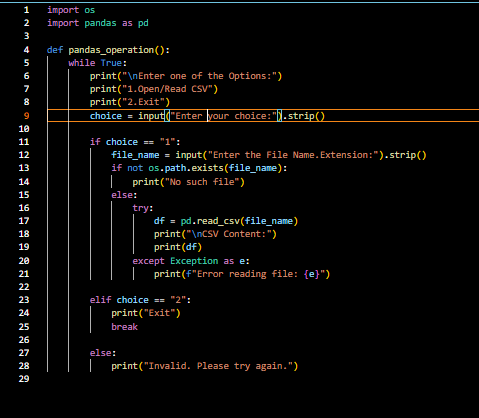
This mode uses the Pandas library to handle CSV files. It includes the following features:

* **Open/Read CSV**:  
  Users are prompted to enter the name of a CSV file. If the file exists, the program reads and displays its contents in a tabular format. If the file is not found or is invalid, the user is informed.
* **Exit**:  
  This option allows users to exit the Pandas operations mode.

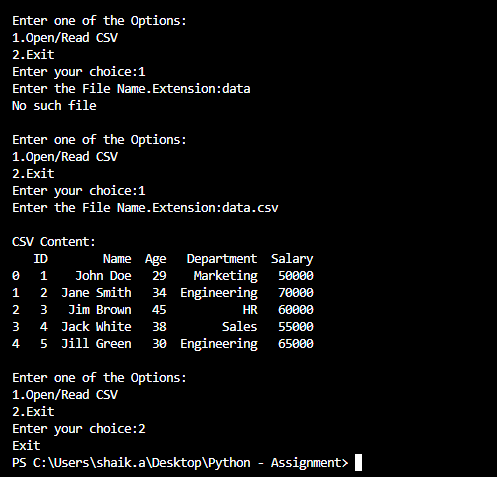
A computer screen shot of a program

Description automatically generated

**Code – Pandas Operation**

****

**Pandas Operation Output**

****

**The key concepts from this solution include:**

1. Modular Design: By separating functionality into different modules (e.g., file\_op and pandas\_op), the program is more maintainable, reusable, and easier to extend**.**
2. User Interaction: The script allows for dynamic interaction based on user input via command-line arguments, making it adaptable for different use cases.
3. Error Handling: Basic error handling is implemented to ensure the program provides helpful feedback in case of missing or invalid inputs, improving the user experience**.**
4. Flexibility: The design ensures flexibility in adding more modes or functionalities in the future, such as adding more file operations or supporting other data formats.