

Overview

In this worksheet, we will explore a Python program that implements Newton's second law of motion. The program prompts the user to input values for mass, acceleration, and force, and calculates the missing value based on the user's input. While entering the user input, the user must leave one of the three values blank to calculate that value. This practical exercise will cover topics such as user input, control structures, and operators.

Eg: If the user wants to calculate the value of force given the value of mass and acceleration, then user should just keep blank for force but enter the value of mass and acceleration.

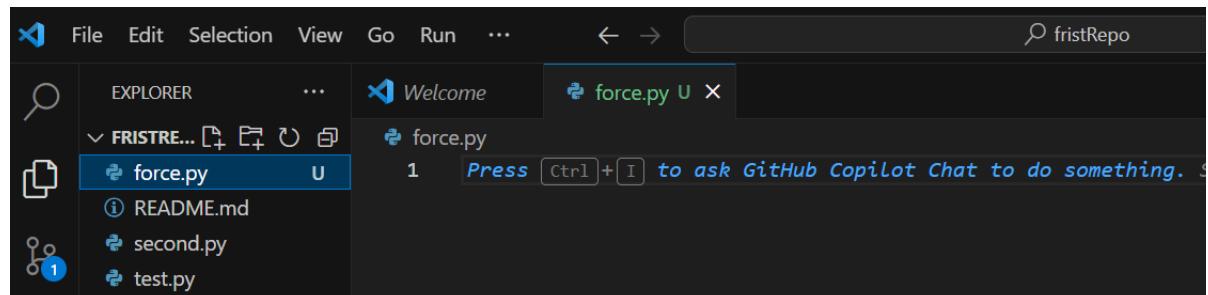
Pre-requisites:

1. Github account
2. Git
3. VSCode
4. Python
5. Basic understanding of Python syntax.
6. Familiarity with variables and control structures(if, else and elif).
7. Knowledge of arithmetic operations and mathematical equations.

Instructions:

1. Open the cloned folder in VSCode.

- a. Create a python file called "force.py"



2. Write a program:

- a. The program starts by printing the header for the program followed by the border line. Type the following print python code in "force.py" file:



- b. Next, the program prompts the user to select the missing value. The user must enter the option number corresponding to the missing value they want to calculate. The program displays three options for the missing value: Mass (m), Acceleration (a), and Force (F). Each option is numbered for user selection. This

input is stored in the variable “missing_value_choice”. Continue typing the following python code in “force.py” file:

```
3
4     # Determine the missing value
5     print("Select the missing value:")
6     print("1. Mass (m)")
7     print("2. Acceleration (a)")
8     print("3. Force (F)")
9     missing_value_choice = input("Enter the option number for the missing value: ")
10
```

- c. Based on the user's input for the missing value, the program executes a specific set of calculations to determine the missing value:
 - i. If the missing value is mass (m), the program prompts the user to input acceleration (a) and force (F), then calculates mass using the formula $m = F / a$. Continue typing the following python code in “force.py” file:

```
11    # Prompt the user to enter the other two values
12    if missing_value_choice == "1":
13        a = float(input("Enter acceleration (a): "))
14        F = float(input("Enter force (F): "))
15        m = F / a
16        print(f"Mass (m) = {m}")
17
```

- ii. If the missing value is acceleration (a), the program prompts the user to input mass (m) and force (F), then calculates acceleration using the formula $a = F / m$. Continue typing the following python code in “force.py” file:

```
18    elif missing_value_choice == "2":
19        m = float(input("Enter mass (m): "))
20        F = float(input("Enter force (F): "))
21        a = F / m
22        print(f"Acceleration (a) = {a}")
```

- iii. If the missing value is force (F), the program prompts the user to input mass (m) and acceleration (a), then calculates force using the formula $F = m * a$. Continue typing the following python code in “force.py” file:

```

23
24     √ elif missing_value_choice == "3":
25         m = float(input("Enter mass (m): "))
26         a = float(input("Enter acceleration (a): "))
27         F = m * a
28         print(f"Force (F) = {F}")
29

```

- iv. If the user enters the option other than the mentioned one, an “invalid option” message will be notified. Continue typing the following python code in “force.py” file:

```

30     else:
31         print("Invalid option selected for the missing value.")
32

```

3. **Run the program:** Now save the program. Open the terminal and then run “force.py” by entering the command “py force.py” or “python force.py” :

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
● PS C:\Users\Tushar\OneDrive\Desktop\New folder (2)\fristRepo> py .\force.py
Newton's Second Law of Motion
-----
Select the missing value:
1. Mass (m)
2. Acceleration (a)
3. Force (F)
Enter the option number for the missing value: 3
Enter mass (m): 5
Enter acceleration (a): 9
Force (F) = 45.0
○ PS C:\Users\Tushar\OneDrive\Desktop\New folder (2)\fristRepo>

```

4. **Push the program file in github:**

i. **Adding, Committing, Checking Status, and Pushing Changes to GitHub**

- Check Status: In the terminal, enter the command “git status” to see that force.py isn’t staged:

```

● PS C:\Users\Tushar\OneDrive\Desktop\New folder (2)\fristRepo> git status
On branch main
Your branch is up to date with 'origin/main'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    force.py

nothing added to commit but untracked files present (use "git add" to track)
PS C:\Users\Tushar\OneDrive\Desktop\New folder (2)\fristRepo>

```

- Stage Changes: Stage the changes by entering the command “git add force.py” in the terminal:

```
p\New folder (2)\fristRepo> git add force.py
```

3. Commit Changes: Commit the staged changes with a commit message entering the command 'git commit -m "Adding force.py file"' in the terminal:

```
PS C:\Users\... \OneDrive\Desktop\New folder (2)\fristRepo> git commit -m "Adding force.py file"
[main f68a7bb] Adding force.py file
 1 file changed, 31 insertions(+)
 create mode 100644 force.py
```

4. Push Changes to GitHub: Push the committed changes to GitHub by entering the command "git push" in the terminal:

```
PS C:\Users\... \OneDrive\Desktop\New folder (2)\fristRepo> git push
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 665 bytes | 332.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/.../fristRepo.git
 4c6ef68..f68a7bb main -> main
```

5. Viewing changes on GitHub

- a. Refresh GitHub Repository Page:

- Goto your GitHub repository page in the web browser.
- Refresh the page to see the changes reflected in the repository. Your "force.py" should get uploaded there.

The screenshot shows a GitHub repository page for 'fristRepo'. The 'Code' tab is selected. The commit history is as follows:

File	Commit Message	Time Ago
README.md	Initial commit	2 days ago
force.py	Adding force.py file	9 minutes ago
second.py	made changes to test.py and adde...	2 days ago
test.py	made changes to test.py and adde...	2 days ago