**LABORATORY EXERCISE 5**

# SIMULATING INFORMATION ARCHITECTURE AND NAVIGATION

# Learning Objectives

* Understand the core principles of information architecture and navigation design in HCI.
* Apply Python programming fundamentals (print, variables, input, conditionals, loops, while).
* Represent a simple information structure (pages, categories) in Python.
* Simulate user navigation through a continuous interactive program.

# Prerequisite student experiences and knowledge

* Students should be familiar with HCI concepts from Modules 1–4.
* No prior programming knowledge is required; this lab introduces Python step by step.

# Background

# Information Architecture (IA) is about organizing and structuring information spaces so users can find and navigate content effectively. Navigation design in HCI ensures that users can move smoothly between sections, similar to wayfinding in a building.

# Python can simulate IA concepts using variables to store information, loops to list menu items, conditionals to handle navigation decisions, and input to allow user interaction. By combining these fundamentals, students can experience how programming directly models navigation and usability.

# Materials/Resources

* Computer with **Python** installed.
* **Visual Studio Code (VS Code)**
* Internet
* Documentation Tools (e.g., Microsoft Word)

**Laboratory Activity**

**Installation Guide**

**Step 1: Install Python**

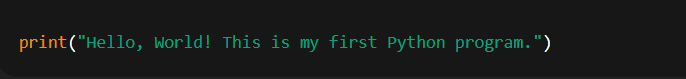
1. Go to the official Python website: <https://www.python.org/downloads/>
2. Click Download Python 3.x (latest version).
3. Open the installer file.
4. ✅ IMPORTANT: Check “Add Python to PATH” before clicking Install Now.
5. Once installed, open Command Prompt (Windows) or Terminal (Mac/Linux) and type:



**Step 2: Install Python Extension in VS Code**

1. In VS Code, click on the Extensions icon (left sidebar or press Ctrl+Shift+X).
2. Search for Python.
3. Install the extension published by Microsoft.

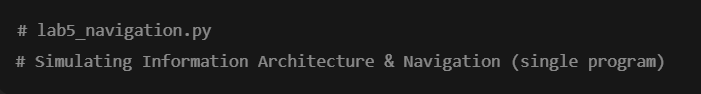
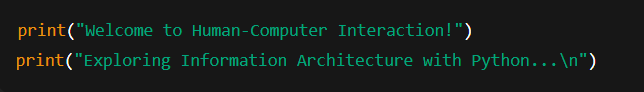
**Step 4: Run Your First Python Program**

1. Open VS Code.
2. Click File > New File and save it as lab5.py.
3. ****Type the following code:
4. Right-click inside the editor and choose Run Python File in Terminal.
5. You should see the output in the terminal.

**INSTRUCTIONS:** Follow the steps carefully to build one complete program.

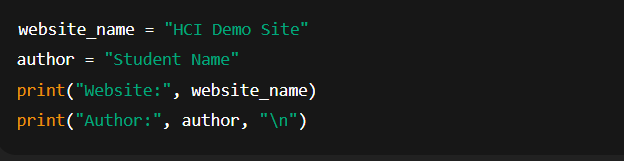
**Step 1 –** Create a new file and save it as **lab5\_navigation.py**.

**Step 2 —** File header & welcome (print)

* In lab5\_navigation.py type:
* Add the welcome/homepage prints:

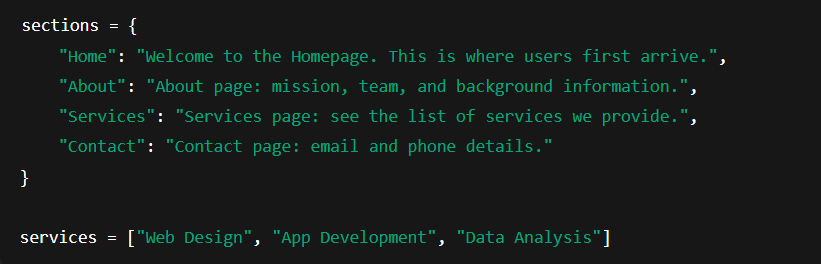
**Why:** print() shows output (visibility / homepage).

**Step 3** — Variables / metadata

* Type the variables that store basic info:

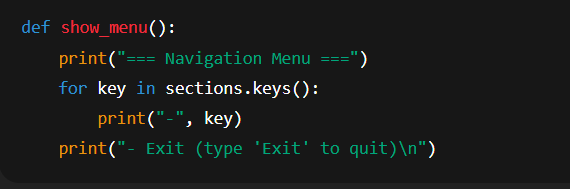
**Why:** Variables hold information about the site (information space metadata).

**Step 4** — Define pages and content (dictionary) and a list for services

* Add the site structure (pages) and a services list:

**Why:** sections models an information architecture (pages → content). services demonstrates grouped content.

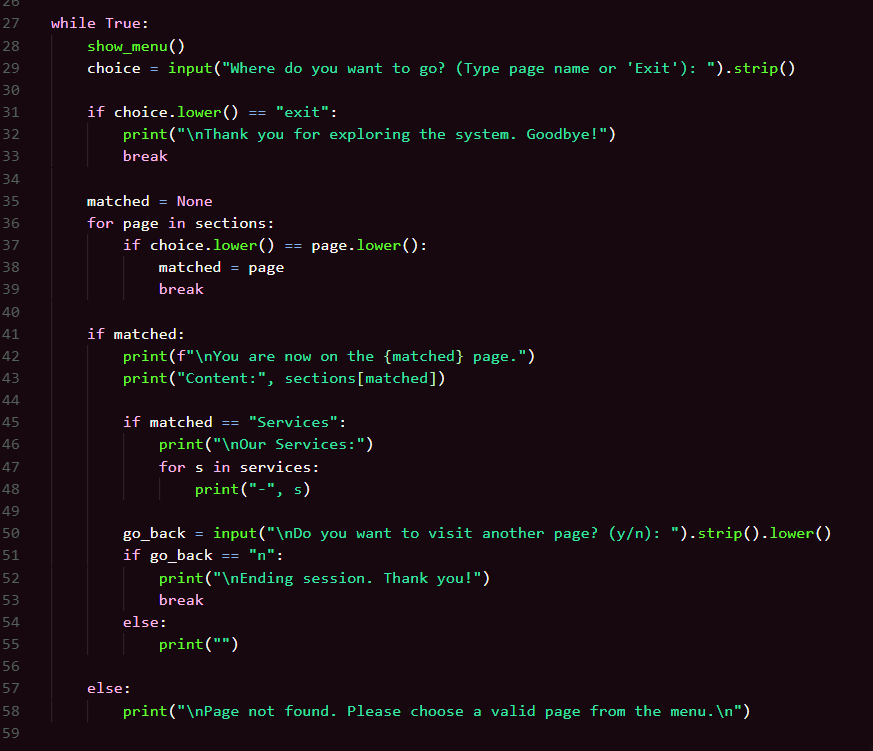
**Step 5** — Menu function

* Create a small function that prints the menu each loop:

**Why:** keeps code organized and shows the explicit navigation bar.

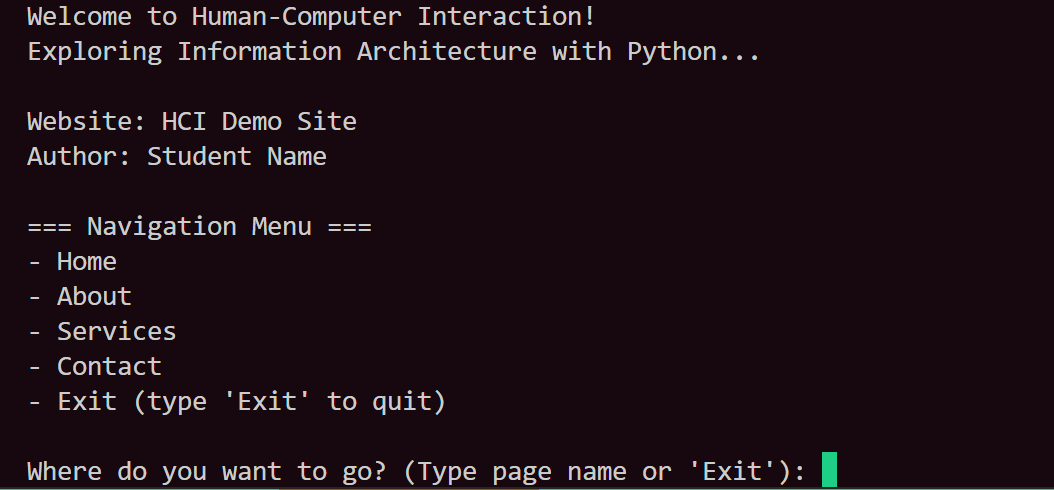
**Step 6** — Main interactive loop (while + input + conditionals + loops)

* Now add the continuous loop that lets the user navigate repeatedly:

**Step 7** — Save and run

* **Save** the file (Ctrl+S).
* Run in VS Code:
* Right-click the editor → **Run Python File in Terminal**, or open Terminal (View → Terminal) and run

**QUICK TEST**

* User runs program → sees menu:
* If the user types Services, they see the list of services.
* If the user types Blog, they see: Page not found. Please choose a valid page from the menu.
* Typing Exit quits the program.

# Question

1. How does this program simulate information architecture and navigation design?
2. What happens when the user inputs an invalid page name? Why is this a usability issue?
3. Suggest one improvement to make this program more user-friendly.

# Output / Results

* Submit your Python file: lab5\_<lastname>.py.
* Include screenshots showing:
  + The menu and a visited page (e.g., Services).
  + An invalid input result (Page not found).

**Conclusion**