Assignment 02

Section 1: HTML & CSS | Build a Restaurant Website

1. Create the Home Page (index.html)

- **Set Up the HTML Structure:**
    1. Create a new file named **index.html**.
    2. Set the title of the page to: "Gourmet Delights - Home".
- **Add a Banner Image:**
    1. Insert an image at the top of the page (700px width) representing the restaurant.
    2. Add **alt text** (e.g., "Gourmet Delights Restaurant").
    3. Float the image to the **center** and style it with a **2px solid black border**.
- **Create the Main Heading and Introduction:**
    1. Add an **H1 heading**: "Welcome to Gourmet Delights!".
    2. Add a **paragraph** describing the restaurant: "We serve delicious food made with fresh ingredients and a passion for cooking!".
- **Create a Navigation Menu with Links:**
    1. Use an **unordered list (ul)** to create a menu with links to the following pages:
        a) **Home** (Link to index.html)
        b) **Our Menu** (Link to menu.html)
        c) **Contact Us** (Link to contact.html)
- **Apply Basic CSS Styling:**
    1. Set the **background color** to lightgoldenrodyellow.
    2. Style the **H1 heading** with **font color: darkred**.
    3. Use the <hr> tag to separate the sections.

2. Create the Menu Page (menu.html)

- **Set Up the HTML Structure:**
    1. Create a new file named **menu.html**.
    2. Set the **title** of the page to: "Gourmet Delights - Our Menu".
- **Create a Header and Menu Categories:**
    1. Add an **H1 heading**: "Explore Our Delicious Menu".
    2. Add three subheadings (H2) for menu categories:
        a) **Appetizers**
        b) **Main Course**
        c) **Desserts**
- **Use Unordered Lists to Display Dishes:**
    1. For each category, add an **unordered list** of dishes (3-4 items per category).
        a) Example for Appetizers:
            1. **Garlic Bread**
            2. **Bruschetta**
            3. **Stuffed Mushrooms**
- **Add a Food Image:**
    1. Insert an image of food (e.g., **400px** width) and float it to the right.
    2. Add **alt text** (e.g., "Delicious Gourmet Dish").
    3. Style the image with a **2px dashed brown border**.
- **Add a Link to the Home Page:**
    1. At the bottom, add a link: "Back to Home" (Link to **index.html**).

3. Create the Contact Page (contact.html)

- **Set Up the HTML Structure:**
    1. Create a new file named **contact.html**.
    2. Set the **title** of the page to: "Gourmet Delights - Contact Us".
- **Add Contact Information:**
    1. Add an **H1 heading**: "Get in Touch with Us!".

2. Add a **paragraph**: "We are here to answer your questions and take reservations."

- **Create a List of Contact Methods:**
  1. Use an **unordered list (ul)** to display:
     a) **Phone:** +123-456-7890
     b) **Email:** contact@gourmetdelights.com
     c) **Address:** 123 Foodie Street, Flavor Town
- **Add a Link to Google Maps:**
  1. Add a hyperlink that says "Find Us on Google Maps" and link it to:
     a) https://www.google.com/maps
- **Add a Link to the Home Page:**
  1. At the bottom, add a link that says "Back to Home" (Link to **index.html**).

4. Additional Requirements:

1. **Ensure Proper HTML Structure:**
   - Use the correct **DOCTYPE declaration** and follow HTML syntax on all pages.
2. **Emphasized Text:**
   - In the **Menu Page**, add a sentence: "We only use the freshest ingredients to bring you the best experience!"
   - Emphasize the word "freshest" using the tag, and style it to appear **bold and green** using inline CSS.
3. **Use Horizontal Lines:**
   - Use the <hr> tag to separate sections on all pages.
4. **Consistent Styling:**
   - Use the same **background color** (lightgoldenrodyellow) on all pages.

Section 2: JavaScript

Create a JavaScript program that allows users to manage a to-do list through a command-line interface. The program should support adding tasks, removing tasks, viewing all tasks, and clearing the list.

Requirements:

1. **Install prompt-sync**:
    - Guide students to install prompt-sync by running npm install prompt-sync in their project directory.
2. **Program Structure**:
    - Define functions for each operation:
        - **addTask():** Adds a task to the list.
        - **removeTask():** Removes a specific task by index.
        - **displayTasks():** Displays all tasks.
        - **clearTasks():** Clears all tasks from the list.
3. **User Interaction**:
    - Prompt the user for commands (add, remove, view, clear, or exit).
    - Based on the command, call the corresponding function.
4. **Error Handling**:
    - Ensure that invalid commands and task indices produce appropriate error messages.

Section 3: CISCO Packet Tracer

1. Create the Network Topology

Use Cisco Packet Tracer to design a network topology that includes:

- 1 Switch
- 5 Computers (PCs)
- 1 Network Printer

2. **Assign IP Addresses**

   Configure the PCs and the printer with IP addresses in the same subnet (e.g., 192.168.1.0/24).

   Make sure to use the following IP addresses:

   - PC1: 192.168.1.2
   - PC2: 192.168.1.3
   - PC3: 192.168.1.4
   - PC4: 192.168.1.5
   - PC5: 192.168.1.6
   - Printer: 192.168.1.10

3. **Configure Network Devices:**
   - Ensure that all computers can ping each other and the printer.
   - Use the command line on each PC to test connectivity (e.g., ping command).
4. **Set Up Printer Sharing:**
   - Configure one of the PCs to act as a print server. Install a virtual printer and ensure that all other PCs can print to it.
5. **Document Your Setup:**
   - Take screenshots of your network configuration in Packet Tracer.
   - Include a brief description of the steps you took to complete the task.

Submission Guidelines:

- Submit your Packet Tracer file (.pkt) with the configured network.
- Include a word document with your screenshots and explanations of each step.

Section 4: LaTeX | Overleaf

1. Create a new Example Project on overleaf.com, by doing so you will get a template for a simple research paper. Your task is to change this and write an example paper about your favorite animal.
2. Make sure to change all the details making them relevant to your topic, such as title, author name, abstract, etc.

3. Include sections, subsections, and paragraphs about scientific information about that animal. Make sure that this information is factual and presented formally.
4. Include at least four relevant images with relevant captions.
5. Include a table with a caption that shows these details about that animal:
    - Scientific name
    - Class (Reptile/ Mammal, etc,..)
    - Eats
6. Cite a research paper about this animal that can be searched on scholar.google.com. Use the bib text of that paper to cite that in your overleaf document.
7. Add a new Section named "Hypothesis about *animal_name_here*"
8. In this section, add a mathematical formula that would calculate the life span of this animal. Formulate the mathematical equation based on your own logic.
9. Save the overleaf as a .pdf for submitting.

Section 5: Object-Oriented Programming – OOP | C++

Create a C++ program that simulates a simple bank account system. The system should allow a user to create an account, deposit money, withdraw money, and check the account balance.

Requirements

1. **Class Definition**:
    - Create a class called **BankAccount.**
    - Define private attributes:
        - **accountNumber (string)** - stores the account number.
        - **balance (double)** - stores the account balance.
2. **Constructor**:
    - Define a constructor that initializes **accountNumber** and sets **balance** to **0**.
3. **Methods**:
    - **deposit(double amount)**: Adds the amount to the current balance.

- **withdraw(double amount):** Checks if there is enough balance, then deducts the amount if possible. Otherwise, it should print "Insufficient balance."
- **checkBalance():** Prints the current balance.

4. **Main Function**:
    - Create an instance of **BankAccount**.
    - Use the object to call **deposit()**, **withdraw()**, and **checkBalance()** methods.