

# Week 3 Homework Assignments: Inputs, Formatted Outputs, and Control Flow

## Global Requirements

- All deliverables shall be added, committed, and pushed to your **Week3** folder in your repository.
- Include your name and the names of anyone who assisted you in the following format:

% Student: Firstname Lastname

% Assisted by: Firstname Lastname, etc.

---

## 1. Basic Beam Load Calculation

### Task

Create a script that calculates the load on a beam based on user input and outputs the results in a formatted manner using **fprintf**.

### Instructions

1. **Create a script** `beam_load.m` that:
  - Prompts the user to input the load applied to the beam (in Newtons) and the length of the beam (in meters).
  - Uses the formula

$$\text{Stress} = \frac{\text{Load}}{\text{Length}}$$

to calculate the stress on the beam.

- Outputs the results using **fprintf** in a clear, formatted way.

Example formatted output: > The load on the beam is 500 N, the length is 3 meters, and the stress is 166.67 N/m.

### Deliverables

1. Submit the script file `beam_load.m`.
  2. Include comments explaining how user input is handled and how **fprintf** is used for formatted output.
- 

## 2. Ingredient Cost Calculator

### Task

Create a script that calculates the total cost of ingredients for a recipe, based on user input. Use a loop to allow multiple ingredients.

### Instructions

1. **Create a script** `ingredient_cost.m` that:
  - Prompts the user to input the cost of each ingredient and the quantity required for a recipe.
  - Uses a **while** loop to allow the user to add as many ingredients as they like.
  - Outputs the total cost using **fprintf** in a formatted way.
  - Use a **while** loop that continues asking for more ingredients until the user indicates they are done.

Example interaction: > Enter the cost of the ingredient: 2.50 > > Enter the quantity required: 3 > > Would you like to add another ingredient? (y/n): y

2. When the user finishes adding ingredients, the script should output the total cost of all ingredients using `fprintf`.

Example formatted output: > The total cost of ingredients is \$15.75.

### Deliverables

1. Submit the script file `ingredient_cost.m`.
  2. Include comments explaining how the loop works and how `fprintf` is used for output.
- 

## 3. Ball Drop Simulation

### Task

Create a script that simulates the free fall of a ball from a given height, using a loop to calculate the position over time.

### Instructions

1. Create a script `ball_drop.m` that:
  - Prompts the user for the initial height of the ball.
  - Simulates the ball falling under gravity ( $9.81 \text{ m/s}^2$ ).
  - Uses a `while` loop to calculate the ball's position every 0.1 seconds until it reaches the ground.
  - Outputs the ball's position at each time step using `fprintf`.

Example interaction: > Enter the initial height of the ball (in meters): 10 > > Time: 0.0 s, Height: 10.00 m  
> > Time: 0.1 s, Height: 9.95 m > > Time: 0.2 s, Height: 9.80 m

2. The loop should stop when the ball reaches the ground (height  $\leq 0$ ).

### Deliverables

1. Submit the script file `ball_drop.m`.
  2. Include comments explaining the use of the `while` loop and `fprintf` for formatted output.
- 

## 4. Bug Hunt Challenge: Common Mistakes in Control Flow and Loops

### Task

Troubleshoot and fix common mistakes related to control flow (`if-else`), loops, and formatted output.

### Instructions

1. Analyze the provided buggy script `buggy_script3.m`:

```
% Check if a number is positive, negative, or zero
num = input('Enter a number: ');
if num > 0
    disp('The number is positive.')
elseif num < 0
    disp('The number is negative.')
else
    disp('The number is zero.')

% Calculate the sum of even numbers between 1 and 100 using a while loop
```

```
% and incrementing i by 2 for each loop iteration.
sum_even = 0;
i = 2;
while i <= 100
    sum_even = sum_even + i;
end
fprintf('The sum of even numbers between 1 and 100 is: %d\n', sum_even);
```

### Deliverables

1. Submit the corrected script `fixed_script3.m`.
  2. Write a short report (`debuggingReport3.txt`) explaining the errors you found and how you fixed them.
- 

### Definition of Done

1. Your `Week3` folder shall contain the following files:
  - `beam_load.m`
  - `ingredient_cost.m`
  - `ball_drop.m`
  - `buggy_script3.m`
  - `fixed_script3.m`
  - `debuggingReport3.txt`