CSE 1321L: Programming and Problem Solving I Lab

Assignment 5

Module 5

Spring 2026

What students will learn

- Problem Solving.
- Basic Program Structure.
- Input and Output with the user.
- Structure program to include conditional logic.
- Write code that includes lists and tuples.

Content

Overview:

For this assignment, you're going to practice making logic in your code. It will include loops and conditional statements with a focus on sequence types. In practical terms, this means you're going to expand on the concepts from previous assignments, but also include things like while loop, for loop statements. Again, start early, practice, and ask a lot of questions.

Final note: *Do not cheat*

If your temptation is to look online, don't. Come see us instead and ask questions – we are here to help. Remember, you are going to have to write codes in your future job interviews, so learn it now to secure a high-paying job later.

Assignment5A: Word Rearranger

Write a Python program that rearranges the words of a sentence based on their length.
The program should prompt the user to enter a sentence containing lowercase words separated by spaces. You must split the sentence into individual words and store them in a **list**.

For this program:

- Your task is to rearrange the words such that:
  - All **short words** (length ≤ 3) appear first
  - All **medium words** (length 4 to 6) appear next
  - All **long words** (length ≥ 7) appear at the end
- The **relative order of words within each category must remain the same** as in the original sentence.
- Display the rearranged sentence as a single line of words separated by spaces.
- Do NOT use nested lists.
- Do NOT use built-in sorting functions.

Example runs are shown below. The user input is shown in **red and bold.**

Sample Output #1:

```
Enter a sentence: I love python programming very much
Rearranged sentence: I love python very much programming
```

Sample Output 2:

```
Enter a sentence: cat elephant dog tiger lion
Rearranged sentence: cat dog tiger lion elephant
```

Assignment5B: Cinema Seat Booking

Write a Python program that simulates booking seats in a movie theater.
The theater should be stored as a **list of lists**, where each inner list represents one row of seats.
Each seat should contain one of the following values:

- **A** for Available
- **R** for Reserved
- **B** for Booked

The user will first enter the number of rows and columns for the theater. Then your program should build the seating chart using random values:

- If a random number is greater than or equal to 0.7, the seat should be marked R
- Otherwise, the seat should be marked A

Your program must keep track of the total number of available seats.
Next, the user should repeatedly enter a row number and a column number to choose a seat. You must store the chosen location as a **tuple**.
When the user selects a seat:

- If the seat is A, mark it as B and tell the user the seat was booked successfully and print seating chart.
- If the seat is R, tell the user it is already reserved.
- If the seat is B, tell the user it has already been booked.

Continue until all available seats have been booked. Then print the final seating chart and a message saying that all available seats have been booked.
Example runs are shown below. The user input is shown in **red and bold.**

Sample Output #1:

```
Enter the number of rows: 2
Enter the number of columns: 2
Number of available seats: 2
R A
A R

Enter the row number (0 to 1): 0
Enter the column number (0 to 1): 0
This seat is already reserved.
R A
A R

Enter the row number (0 to 1): 0
Enter the column number (0 to 1): 1
Seat booked successfully!
R B
A R

Enter the row number (0 to 1): 0
Enter the column number (0 to 1): 1
This seat has already been booked.
R B
A R

Enter the row number (0 to 1): 1
Enter the column number (0 to 1): 0
Seat booked successfully!
```

```
R B
B R
All available seats have been booked!
R B
B R
```

Sample Output 2:

```
Enter the number of rows: 3
Enter the number of columns: 3
Number of available seats: 2
R A R
R R R
R R A

Enter the row number (0 to 2): 0
Enter the column number (0 to 2): 0
This seat is already reserved.
R A R
R R R
R R A

Enter the row number (0 to 2): 0
Enter the column number (0 to 2): 1
Seat booked successfully!
R B R
R R R
R R A

Enter the row number (0 to 2): 2
Enter the column number (0 to 2): 2
Seat booked successfully!
R B R
R R R
R R B
All available seats have been booked!
R B R
R R R
R R B
```

Assignment5C: Matrix Multiplication

Write a Python program that takes input for two matrices, validates whether matrix multiplication is possible, and if valid, displays both input matrices and their resulting product matrix.
All matrix values must be entered by the user as comma-separated values on a single line. The program must internally divide these values into rows and columns based on the dimensions entered by the user.

For this task:

- Your solution must use nested loops for:
  - Dividing input values into rows and columns
  - Displaying matrices
  - Computing matrix multiplication
- Assume the user will input positive integers for matrix dimensions and values.
- The program must prompt the user to enter:
  - Number of rows and columns for Matrix A
  - Number of rows and columns for Matrix B
- Before taking matrix values, the program must **validate** whether matrix multiplication is possible: **Multiplication is possible only if the number of columns in Matrix A is equal to the number of rows in Matrix B.**
- If multiplication is not possible:
  - Display an appropriate error message
  - Terminate the program
- If multiplication is possible:
  - Prompt the user to enter all values of Matrix A on a single line, separated by commas
  - Prompt the user to enter all values of Matrix B on a single line, separated by commas
  - **Assume users will enter valid input matrices here**.
- The program must:
  - Divide the comma-separated values into the correct rows and columns
  - Display Matrix A in row-column format
  - Display Matrix B in row-column format
  - Compute and display the resulting matrix
- Do **not** use built-in matrix or numerical libraries.

Hint:

- Matrix multiplication follows these rules:
  - If Matrix A has dimensions $\mathbf{R_1 \times C_1}$
  - If Matrix B has dimensions $\mathbf{R_2 \times C_2}$
- Multiplication is valid only when:
  - $C_1 == R_2$
- The resulting matrix will have dimensions:
  - $R_1 \times C_2$
- Each value in the resulting matrix is computed by:
- Multiplying corresponding elements from a row of Matrix A and a column of Matrix B
- Summing those products using a nested loop
- More details about matrix multiplications can be found on this link.

Example runs are shown below. The user input is shown in **red and bold.**

Sample Output #1:

```
Enter rows for Matrix A: 2
Enter columns for Matrix A: 2

Enter rows for Matrix B: 2
Enter columns for Matrix B: 2

Enter Matrix A values (comma-separated):
1,2,3,4

Enter Matrix B values (comma-separated):
5,6,7,8

Matrix A:
1 2
3 4

Matrix B:
5 6
7 8

Resulting Matrix:
19 22
43 50
```

Sample Output 2:

```
Enter rows for Matrix A: 5
Enter columns for Matrix A: 3

Enter rows for Matrix B: 5
Enter columns for Matrix B: 3

Matrix multiplication is not possible.
Number of columns in Matrix A must equal number of rows in Matrix B.
```

Sample Output 3:

```
Enter rows for Matrix A: 2
Enter columns for Matrix A: 3

Enter rows for Matrix B: 3
Enter columns for Matrix B: 2

Enter Matrix A values (comma-separated):
1,2,3,4,5,6

Enter Matrix B values (comma-separated):
7,8,9,10,11,12
```

```
Matrix A:
1 2 3
4 5 6

Matrix B:
7 8
9 10
11 12

Resulting Matrix:
58 64
139 154
```

Submission Instructions:

- Programs must follow the output format provided. This includes each blank line, colons (:), and other symbols.
- Programs must be working correctly.
- Programs must be written in Python.
- Programs must be submitted with the correct **.py** format.
- Programs must be saved in files with the correct file name:
  - Assignment5A.py
  - Assignment5B.py
  - Assignment5C.py
- Programs (source code files) must be uploaded to Gradescope by the due date.