YORK UNIVERSITY
LASSONDE SCHOOL OF ENGINEERING
DEPARTMENT OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE

EECS 2032E - Introduction to Embedded Systems

Summer 2025

Lab 5

Due Date: June 14, 2025

Lab Objectives

- To write *simple (beginner level)* `C` programs using conditional statements, loops, functions, arrays, and pointers.

Note:

- The red text is what you type and the blue text is the computer response.
- Save all the `.c` files in a folder named **Lab5**. Compress the folder into a ZIP file and submit it on eClass.

Pre-Lab

Review the course slides of Week 5. You may need to refer the course slides of Week 4 if needed.

Problem 1

Write a `C` program that generates a right-angle triangle filled with asterisks (`*`) and displays it on the monitor. The triangle should be aligned based on a specified left margin for the first row, with each subsequent row having one more asterisk and starting further to the left. The program takes two integer inputs from the user, `l` and `m`, where `l` represents the number of spaces from the left edge of the monitor to the first row of the triangle and `m` represents the number of rows in the triangle. The number of asterisks (`*`) in the first row will be `1`, and each subsequent row will contain one additional asterisk.

The program should print the triangle row by row. The number of spaces at the beginning of each row decreases by one as you move down the triangle, ensuring correct alignment. For example, given `l = 5` and `m = 6`, the output should look like:

```
     *
    **
   ***
  ****
 *****
******
```

The program must prompt the user for the values of `l` (left margin for the first row) and `m` (number of rows). The triangle should be printed according to the alignment specified by the input, with each row containing one more asterisk than the previous row. Ensure that the left margin decreases by `1` for each subsequent row.

Hints

- Use nested loops to handle the number of spaces and asterisks in each row.
- The outer loop controls the rows, while the inner loops handle spaces and asterisks, respectively.

Save the file as `lab5_1.c`

Problem 2

In this `C` program, we will pass an array with 6 integers to a function, which will swap the first and last integer, the second and the second to last integer, the third and the third to last integer.

The function is called `backwardArray` and doesn't return anything `(void)`. It should take one parameter, representing the array of integers.

The main function first reads `6` integers from the input, and assigns them to the array. The main function then calls `backwardArray`, passing the array as an argument.

The `main` function then prints the backward (or reversed) array.

Example 1

Input

```
1 2 3 4 5 6
```

Output

```
6 5 4 3 2 1
```

Example 2

Input

```
9 12 3 25 11 5
```

Output

```
5 11 25 3 12 9
```

Save the file as `lab5_2.c`

Problem 3

You have developed a powerful "Reversal Potion" that can either reverse or enhance specific physical characteristics based on a person's age. Before buying the potion, customers want to know what effect the potion will have on them!

- If a person is 25 years old or younger, the potion increases their strength by a factor of 2.
- If a person is 26 to 40 years old, the potion reduces their weight by 10 percent.
- If a person is 41 years old or older, the potion boosts their wisdom by 5 points.

You need to write a `C` program that defines a `main()`, which takes the input from the user and then calls a function named `applyPotion`. This function should accept four pointers - one pointing to an integer representing the age, one pointing to an integer representing strength, one pointing to a float representing weight, and one pointing to an integer representing wisdom. The function should modify these values according to the rules of the potion. It should have a `void` return type; all modifications should be done via the pointers. The `main()` function then prints the modified values on the standard output.

Example 1

Input

```
Enter age:  23
Enter strength level:  20
Enter weight:  70.0
Enter wisdom level:  12
```

Output

```
After drinking the Reversal Potion:
Age:  23
```

```
Strength:  40
Weight:  70.00
Wisdom:  12
```

Example 2

Input

```
Enter age:  35
Enter strength level:  25
Enter weight:  80
Enter wisdom level:  15
```

Output

```
After drinking the Reversal Potion:
Age:  35
Strength:  25
Weight:  72.00
Wisdom:  15
```

Example 3

Input

```
Enter age:  45
Enter strength level:  18
Enter weight:  65.5
Enter wisdom level:  30
```

Output

```
After drinking the Reversal Potion:
Age:  45
Strength:  18
Weight:  65.5
Wisdom:  35
```

Save the file as `lab5_3.c`

Problem 4

Write a `C` program that reads two integers, `n` (rows) and `m` (columns), followed by an `n` × `m` matrix stored in a 1D array (row-major order). The program should identify and print the 1-based indices of rows where all elements are in non-descending order.

Constraints:

- $1 \leq n, m \leq 50$
- Use pointer arithmetic.

Input Format:

- Two integers: `n m` (number of rows and columns).
- `n` $\times$ `m` integers representing the matrix.

Input Format:

- Print the 1-based row indices where elements are in non-descending order, each in `%5d` format on a new line.
- If no such row exists, print nothing.

Example

Input

```
3 4
1 2 3 4
5 3 7 8
2 2 2 2
```

Output

```
    1
    3
```

Save the file as `lab5_4.c`