I. Essay (45%)

1. **[LO 1 & LO 2, 15 points]** Jaja, Lala, and Bebe is developing a robot with thoughts that they can conquer the earth with it in the future. For the time being, they are still in the early stage of the development. They are currently testing on the movement algorithm for the robot. The robot will **receive a sets of input**, **reorder** it **based on level of importance**, then **perform the movement** as ordered. Please help them simulate the robot movement. The following are the criteria of the robot movement:

a) The robot can only receive command: $\boldsymbol{move}(\boldsymbol{levelOfImportance}, \boldsymbol{direction})$

b) There are **four level of importance**, **ordered by the most importance** are **CRITICAL**, **HIGH**, **NORMAL**, **LOW**. If there are multiple order with same importance level, then the earlier input order must be executed first (first come first serve).

c) **Direction** the robot can move are: **up**, **down**, **left** and **right**. Each type of movement on

d) The robot will move in a **Cartesian Diagram**. The robot position will be indicated with a coordinate in the form of $(\boldsymbol{x}, \boldsymbol{y})$, where $\boldsymbol{x}$ is the coordinate along the **$\boldsymbol{x}$-axis** and $\boldsymbol{y}$ is the coordinate along the **$\boldsymbol{y}$-axis**. The **robot starting position** is always $(\mathbf{0}, \mathbf{0})$.

e) The diagram **does not have negative value**, so any movement that makes the robot goes to **negative $\boldsymbol{x}$ or $\boldsymbol{y}$** value **will be ignored**.

Example of input command:

Input Command	Ordered Input Based on Level of Importance

`move(HIGH, "up")`
`move(LOW, "right")`
`move(CRITICAL, "down")`
`move(LOW, "left")`
	`move(CRITICAL, "down")`
	`move(HIGH, "up")`
	`move(LOW, "right")`
	`move(LOW, "left")`

Therefore, with starting point of $(0, 0)$, and movement order of "down", "up", "right", then "left", the robot will ends up at coordinate $(0, 1)$.

Based on information above, please simulate the robot movement for the following input (the movement order), then determine which coordinate is the robot final position!

- move(HIGH, "up")
- move(LOW, "right")
- move(NORMAL, "up")
- move(CRITICAL, "up")
- move(CRITICAL, "up")
- move(HIGH, "down")
- move(CRITICAL, "up")

- move(NORMAL, "left")
- move(NORMAL, "right")
- move(HIGH, "up")
- move(CRITICAL, "left")
- move(LOW, "right")
- move(LOW, "up")
- move(HIGH, "right")
- move(LOW, "down")

2. **[LO 1 & LO 2, 15 points]** Given the following mathematical statement in the form of Infix:

$$(\mathbf{A} - (\mathbf{B} * (\mathbf{C} + \mathbf{D})) - \mathbf{E} / \mathbf{F} \wedge \mathbf{G})$$

using Stack Method, please do:

a. Convert those statements into Postfix.

b. Evaluate the result.

Please use the given table as reference for each operator precedence (lower means least prioritized). For same precedence level operator, please use first come first serve rules (left to right).

Operator	**Notes**	**Precedence**
+	Addition	1
-	Subtraction	1
*****	Multiplication	2
/	Division	2
^	Exponentiation	3

3. Given a hash table with size 4, started with index 0 to 3. You are tasked with hashing a list of strings with length 4 with following orders:

tzfo, nkkl, lqmk, xoji

There is several information you need:

a. Hash function used is folding with custom criteria:
   - Convert each character of string to a string of two digits number based on its position alphabetically, starting with index 0 until 25. ‘a’ will be in position 0. ‘z’ will be in position 25. Therefore, ‘a’ will converted to “00”, while z will convert into “26”
   - Append each of the converted character into hash value. For example if we have string “abz”, this will be converted to “000125”
   - Do folding for the hash value with 1 length digit (for string 000125 will be divided to 0 0 0 1 2 and 5).

b. Collision mitigation using linear probing.

Write your hashing process for each of these strings and the final hash table.

II. Case (55%)

[LO 3, 55 Points]

Mr. X, a man with a gaming hobby, has many games. Because he has too many games to track, he decided to store the game names that he owns. To make searching efficient, he decides to use BST as the storing system.

He decides that that BST has some properties:

- Each node in BST stores **game name** and **game rating (1 – 10).**
- BST will sort the data based on **game name** (string) **alphabetically in DESCENDING ORDER** and **case insensitive**.

Here are the operations that can be applied:

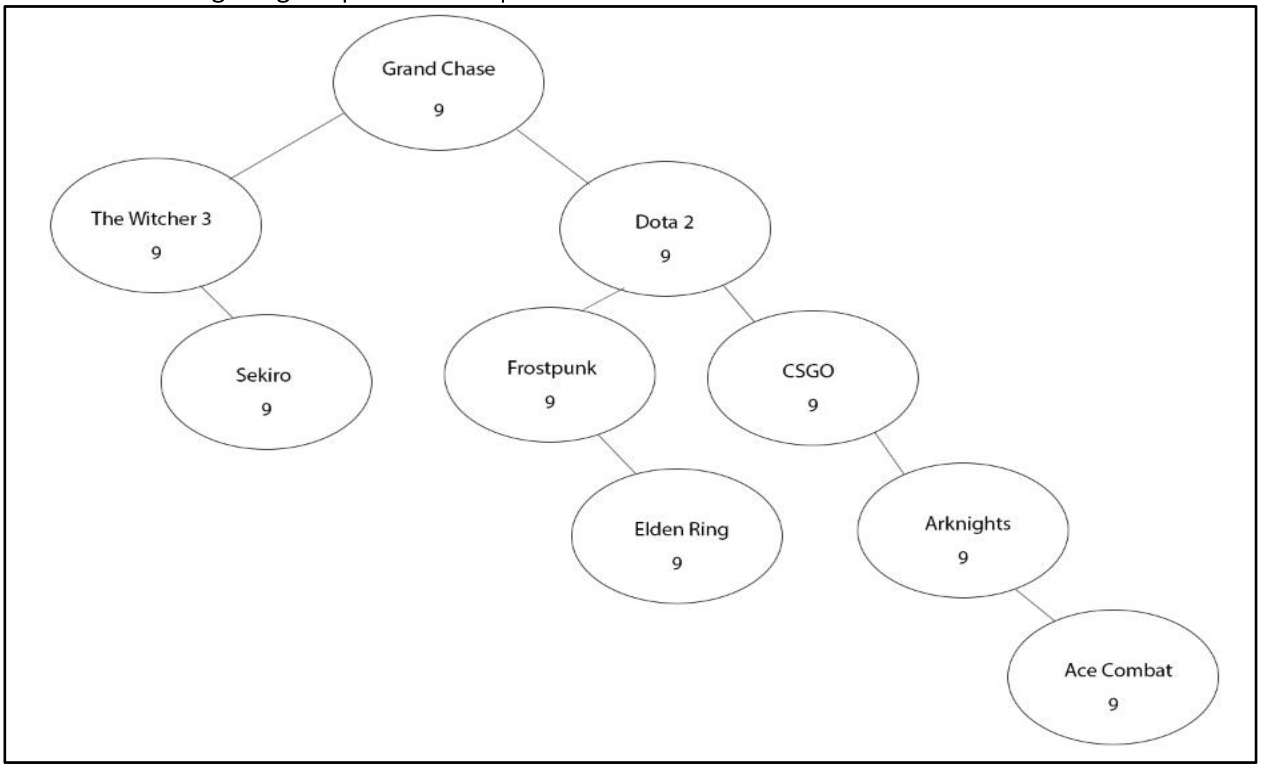
1. Insert

   If X chooses this menu, the program will request **game name** and **game rating**. If **there is an existing game** name in BST, **update** the rating instead.
   - Validate the game name has **minimum length of 1 and maximum length of 100**.
   - Validate the game rating value has **range between 1 and 10.**

2. Delete

   If X chooses this menu, the program will inquire about the **game name** to be deleted.
   - Validate the game name has **minimum length of 1 and maximum length of 100**.
   - If the game name is **NOT found**, display the message “Game name does not exist!”.
   - If the game name is **found**, delete that node using the BST deletion method.

The following image depicts the sample BST Mr. X has in mind:



Write a .C console program to help Mr. X !

--Good Luck--