Exercise – Queues

1. Create a queue class using the layout provided in the slides. It should contain the following functionality:
    a) push(value) – Pushes a value onto the back of the queue
    b) pop() – Removes the element at the front of the queue
    c) top() - Get the element at the front of the queue
    d) empty() - Returns true if the queue is empty
    e) size() - Returns the number of elements in the queue

2. You are now going to use your queue class to simulate a server handling messages sent to it by a client. Due to CPU constraints, your server can only handle a maximum of 5 messages per frame, but currently it has no limit.

    First, download the *AIE Year1 Samples* repository from GitHub: https://github.com/AcademyOfInteractiveEntertainment/AIEYear1Samples. It contains the *CDDS_Queues_TreasureHunt* project that has a Server that is queried for messages that specify directions of movement for a collection of "treasure hunters". The application also defines a "Treasure Map" that stores the location of the treasure. The server moves the treasure hunters around and you job is to collect and process these messages, exiting the program once one of the treasure hunters reaches the location of the treasure.

    Each time you process movement you should update the treasure hunter's position variables.

    Using the example code in TreasureHunt.cpp and using your queue class, add functionality to the TreasureHunt.cpp main function to allow it to queue messages if there are more than 5 messages received from the client in a single frame. Your application's output should look similar to the output below:

    Size of Message Queue before Retrieving Messages: 8

    ...

    Size of Message Queue after Retrieving Messages: 3

    Size of Message Queue before Retrieving Messages: 3

    ...

    Size of Message Queue after Retrieving Messages: 0

    …

    Player 3 found the treasure at position 61, 22!