# CSE 3010 – Data Structures & Algorithms

**Lecture #24**

# What will be covered today

- Mid-semester examination pattern
- Implementation of queue data structure using doubly linked list

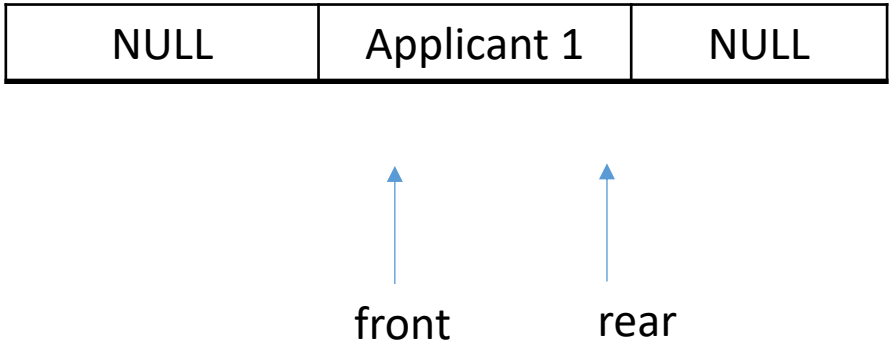
# Mid-semester examination pattern

| | |
|---|---|
| Date of Examination | 27th January 2020 |
| Time | 9:00 am |
| Duration | 2 hours |
| Format | Open book |
| Type | Problem solving |
| Through | LMS |
| Syllabus | Until **Queue data structure** |

# Implementation of Queue using doubly linked list


`front` and `rear` pointers of a queue are null when a queue is created
`front = rear = NULL` [also signifies empty queue)

When the first node is added:



# Creating a node and a new queue

```
QNODE* createNode(ITEM item) {
    QNODE *temp = (QNODE*) malloc(sizeof(QNODE));
    temp->item = item;
    temp->prev = NULL;
    temp->next = NULL;
    return temp;
}

QUEUE* createQueue() {
    QUEUE *queue = (QUEUE*) malloc(sizeof(QUEUE));
    queue->front_queue = NULL;
    queue->rear_queue = NULL;
    return queue;
}
```

# Add an item to a queue using doubly linked list

```
int add(QUEUE *queue, ITEM item) {
    QNODE *temp = createNode(item);
    if (!isFull(queue)) {
        if (isEmpty(queue)) {
                        queue->front_queue = temp;
                        queue->rear_queue = temp;
        }
        else {
                temp->prev = queue->rear_queue;
                        queue->rear_queue->next = temp;
                        queue->rear_queue = temp;
        }
        return 1;
     }
     else
        return 0; // When Queue is full
}
```

# Delete an item from a queue using doubly linked list

```
ITEM delete(QUEUE *queue) {
    ITEM tempItem;
    if (isEmpty(queue))
        tempItem.appln_name[0] = '\0';
    else {
        QNODE *tempNode;
        tempNode = queue->front_queue;
        tempItem = queue->front_queue->item;
        queue->front_queue = queue->front_queue->next;
        if (queue->front_queue != NULL)
            queue->front_queue->prev = NULL;
        free(tempNode);
        if (queue->front_queue == NULL)
            queue->rear_queue = NULL;
    }
    return tempItem;
}
```