**Lab Exercises 6: Queues**

**LEARNING OUTCOMES**

* Understand the problem related to queue
* Able to implement queue using linked list

### PROBLEM : ENQUEUE & DEQUEUE USING LINKED LIST

**LAB EXERCISE 6.1**

Estimate Time**: *2 Hours***

**Tasks:**

Based on the main program in Figure 6.3, write three functions *enqueue ( ), dequeue ( )* and *list\_node( )* as anillustration of queue using linked list. Use the algorithm in Figure 6.1 for *enqueue ( )* andalgorithm in Figure 6.2 for *dequeue ( )*.

1. Create a new node.
2. Enter the new node information and set the link field of the new node to NULL.
3. If FRONT is NULL
   1. FRONT point to NEWNODE
   2. REAR point to NEWNODE
4. If FRONT is not NULL
   1. Link field of REAR point to NEWNODE
   2. REAR point to NEWNODE

Figure 6.1: Algorithm for *enqueue ()*

1. TEMP point to FRONT
2. If FRONT is NULL
   1. List is empty
3. If FRONT is not NULL
   1. FRONT point to link field of FRONT
   2. Delete TEMP

Figure 6. 2: Algorithm for *dequeue ()*

*struct node*

*{ int element;*

*struct node \*ptrnext;};*

*struct node \*front, \*rear,\*thisptr, \*newptr;*

*void main()*

*{*

*char ch;*

*int choice=TRUE;*

*front=NULL;*

*while(choice==TRUE)*

*{ printf("\n\n[-----------Queue Menu-----------]");*

*printf("\nA - Enqueue queue");*

*printf("\nB - Dequeue queue");*

*printf("\nD - Display queue content");*

*printf("\nX - Exit\n");*

*printf("\nEnter choice: ");*

*scanf(" %c",&ch);*

*ch=toupper(ch);*

*switch(ch)*

*{*

*case 'A':enqueue();break;*

*case 'B':dequeue();break;*

*case 'D':list\_node();break;*

*case 'X': choice=FALSE; break;*

*default: printf("\nOnly one of the above");*

*}*

*}*

*getch();*

*}*

Figure 6.3: Main Program