EENG212/INFE212: Algorithms and Data Structures
Fall 2013-2014, **Date**: 30th Dec. 2013, **Duration**: 30 min
Lecturer: Prof. Dr. Hasan Amca
Quiz 5

Student No:

Name:

```
/* Implementation of Queue using Linked List. The program has 4 options.
 * 1-Insert a Node into the Queue,
 * 2-Delete a Node from the Queue,
 * 3-Display all Nodes in the Queue,
 * 4-Exit */
#include <stdio.h>
#include <stdlib.h>

typedef struct node
{       int data;
        struct node *link;
} NODE;          // NODE IS AN ALIAS (shorthand writing) of struct node.
                 // In other words, NODE is now a new data type, just as int
void Insert(int);
int Delete();
void Display();

NODE *head, *tail; // *head and *tail are the 2 variables of type NODE

main()
{
        int option, data;
        head = tail = NULL;
        do
        {       printf("LINKED-LIST IMPLEMENTATION OF QUEUE OPERATIONS\n\n");
                printf("   PLEASE CHOOSE YOUR OPRION\n");
                printf("==============================\n");
                printf("1-INSERT A NEW NODE\n");
                printf("2-DELETE AN EXISTING NODE\n");
                printf("3-DISPLAY ALL EXISTING NODES\n");
                printf("4-EXIT THE PROGRAM\n");
                        printf("\n\nPlease Enter Your Choice: ");
                        scanf("%d", &option);
                        printf("\n\n");
                switch (option)
                { case 1:
                        printf("\n\nREAD THE DATA TO BE INSERTED?");
                        scanf("%d", &data);
                        Insert(data);
                        break;
                case 2:
                        data = Delete();
                        if (data != -1)
                        printf(" A NODE IS DELETED FROM head. THE DELETED VALUE IS: %d\n", data);
                        break;
                case 3:
                        printf("LINKED-LIST IMPLEMENTATION OF QUEUE:\nCURRENT Q STATUS IS:\n");
                        Display();
                        break;
                case 4:
                        printf("\nTerminating\n");
                        break;
                default:
                        printf("\n\nInvalid Option !!! Try Again !! \n\n");
                        break;
                }
                printf("\n\n\n\n  Press a Key to Continue . . . ");
                getchar();
        } while (option != 4);
}

void Insert(int info)
{Please write your function definition on the back of this paper}

int Delete()
{       int info;
        NODE *t;
        if (head == NULL)
        {
                printf(" Underflow!!!");
                return -1;
        }
        else
        {
                t = head;
                info = head->data;

                if (head == tail)
                        tail = NULL;
                head = head->link;
                t->link = NULL;
                free(t);
                return (info);
        }
}

void Display()
{       NODE *t;
        if (head == NULL)
                printf("Empty Queue\n");
        else
        {       t = head;
                printf("\n    head->");

                while (t)                {
                        printf("[%d]->", t->data);
                        t = t->link;     }
                printf("tail\n");
        }
}
```

Question

The program given is a menu driven Queue implementation program using linked list. All the parts of the program are given except the function for entering new nodes into the linked list. Please write, in the back of the page, the missing source code for the function definition

```
void Insert(int);
```

```
void Insert(int info)
{









}
```

Solution

```
/* Implementation of Queue using Linked List.
 * The program has four options, which are:
 *
 * 1-Insert a Node into the Queue,
 * 2-Delete a Node from the Queue,
 * 3-Display all Nodes in the Queue,
 * 4-Exit
 */
#include <stdio.h>
#include <stdlib.h>

typedef struct node
{
	int data;
	struct node *link;
} NODE;			// NODE IS AN ALIAS (shorthand writing) of struct node.
				// In other words, NODE is now a new data type, just as int

// THESE ARE THE FUNCTION PROTOTYPES TO INSERT, DELETE AND DISPLAY NODES
void Insert(int);
int Delete();
void Display();

NODE *head, *tail; // *head and *tail are the 2 variables of type NODE

main()
{
	int option, data;
	head = tail = NULL;

	do
	{
		printf("LINKED-LIST IMPLEMENTATION OF QUEUE OPERATIONS\n\n");
		printf("   PLEASE CHOOSE YOUR OPRION\n");

	printf("==============================\n");
		printf("1-INSERT A NEW NODE\n");
		printf("2-DELETE AN EXISTING NODE\n");
		printf("3-DISPLAY ALL EXISTING NODES\n");
		printf("4-EXIT THE PROGRAM\n");
			printf("\n\nPlease Enter Your Choice: ");
			scanf("%d", &option);
			printf("\n\n");
		switch (option)
		{
		case 1:
			printf("\n\nREAD THE DATA TO BE INSERTED?");
			scanf("%d", &data);
			Insert(data);
			break;
		case 2:
			data = Delete();
			if (data != -1)
				printf(" A NODE IS DELETED FROM head. THE DELETED DATA VALUE IS: %d\n", data);
			break;
		case 3:
			printf("LINKED-LIST IMPLEMENTATION OF QUEUE OPERATIONS:\nCURRENT QUEUE STATUS IS:\n");
			Display();
			break;
		case 4:
			printf("\nTerminating\n");
			break;
		default:
			printf("\n\nInvalid Option !!! Try Again !! \n\n");
			break;
		}

		printf("\n\n\n\n  Press a Key to Continue . . . ");
		getchar();
	} while (option != 4);
}

void Insert(int info)
{
	NODE *temporary_pointer;
	temporary_pointer = (NODE *) malloc(sizeof(NODE));

	if (temporary_pointer == NULL)
		printf(" Out of Memory !! Overflow !!!");
	else
	{
		temporary_pointer->data = info;
		temporary_pointer->link = NULL;

		if (head == NULL)
		{
			head = tail = temporary_pointer;
		} /* First Node? */
		else
		{
			tail->link = temporary_pointer;
			tail = temporary_pointer;
		} /* Insert End */

		printf(" Node has been inserted at End Successfully !!");
	}
}

int Delete()
{
	int info;
	NODE *t;

	if (head == NULL)
	{
		printf(" Underflow!!!");
		return -1;
	}
	else
	{
		t = head;
		info = head->data;

		if (head == tail)
			tail = NULL;
		head = head->link;
		t->link = NULL;
		free(t);
		return (info);
	}
}

void Display()
{
	NODE *t;

	if (head == NULL)
		printf("Empty Queue\n");
	else
	{
		t = head;
		printf("\n    head->");

		while (t)
		{
			printf("[%d]->", t->data);
			t = t->link;
		}
		printf("tail\n");
	}
}
```