PRESIDENCY UNIVERSITY, BENGALURU
SCHOOL OF ENGINEERING

Max Marks: 40 | Max Time: 60mins | Weightage: 20 %

TEST 2

I Semester AY 2017-18 | Course: **CSE 201 Data Structure** | 27 OCT 2017

Instructions:

i. Write legibly
ii. Scientific and non programmable calculators are permitted

---

Part A

(3Q x 3 M= 09 Marks)

1. Define queue with an example. Write any two real-time applications of queue.
2. Write the differences between array and linked list. List the operations in Singly Linked List.
3. Define Header node. What is the difference between Singly Linked List and Doubly Linked List with respect to the node structure?

Part B

(2Q x 8 M= 16 Marks)

4. Write a function to delete any duplicate nodes from a singular linked list sorted in non-decreasing order. The list should only be traversed once.
5. Given a singly linked list, write a function to swap elements pairwise.
   Sample Input 1->2->3->4->5 Output : 2->1->4->3->5,

   1->2->3->4->5->6 Output : 2->1->4->3->6->5.

Part C

6. a . Define Circular linked list? Explain the operations on circular linked list. Write functions to insert and delete a node at any given position.

(10 Marks)

b. Write an algorithm to implement the function enque() for a QUEUE using arrays.

(5Marks)

Solutions with scheme

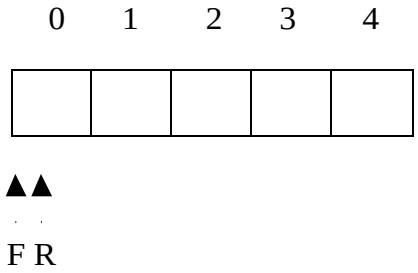
PART-A

1. Define queue with example. Write any two real-time applications of queue.

A queue is another special kind of list, where items are inserted at one end called the rear and deleted at the other end called the front. Another name for a queue is a "FIFO" or "First-in-first-out" list. -1 Mark

Example: -1 Mark

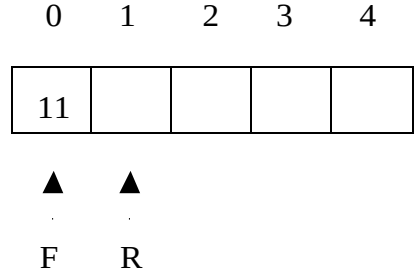
Let us consider a queue, which can hold maximum of five elements. Initially the queue is empty.



Queue Empty

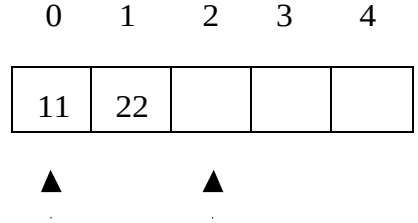
F RO NT = RE A R = 0

Now, insert 11 to the queue. Then queue status will be:



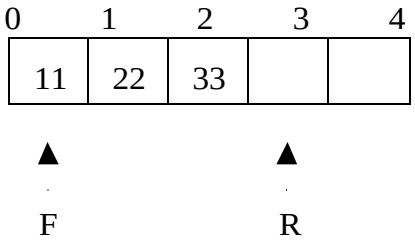
RE A R = RE A R + 1
= 1 FRO NT = 0

Next, insert 22 to the queue. Then the queue status is:

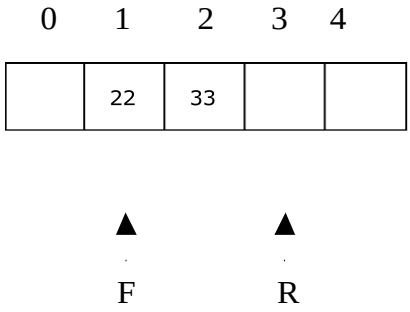


RE A R = RE A R + 1
= 2 FRO NT = 0

Again insert another element 33 to the queue. The status of the queue is:



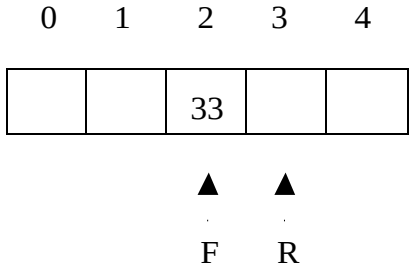
Now, delete an element. The element deleted is the element at the front of the queue. So the status of the queue is:



RE A R = 3

F RO NT = F R O NT + 1 = 1

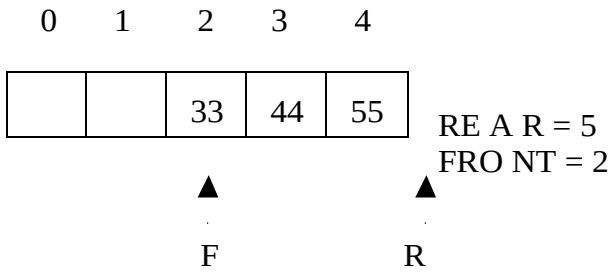
Again, delete an element. The element to be deleted is always pointed to by the FRONT pointer. So, 22 is deleted. The queue status is as follows:



RE A R = 3

F RO NT = F R O NT + 1 = 2

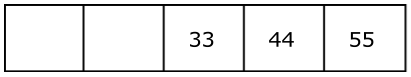
Now, insert new elements 44 and 55 into the queue. The queue status is:



RE A R = 5

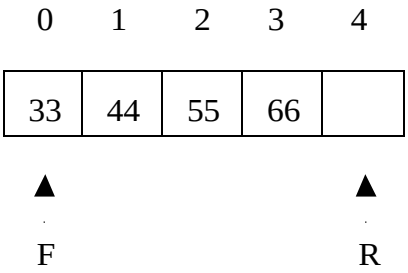
FRO NT = 2

Next insert another element, say 66 to the queue. We cannot insert 66 to the queue as the rear crossed the maximum size of the queue (i.e., 5). There will be queue full signal. The queue status is as follows:



0 1 2 3 4

Now it is not possible to insert an element 66 even though there are two vacant positions in the linear queue. To over come this problem the elements of the queue are to be shifted towards the beginning of the queue so that it creates vacant position at the rear end. Then the FRONT and REAR are to be adjusted properly. The element 66 can be inserted at the rear end. After this operation, the queue status is as follows:



Applications: 1) Ticket Counter 2) Processes executed by operating system -1 Mark

2. Write the differences between array and linked list. List the operations in Singly Linked List.

Solution:

Differences between array and linked list: (Any 4) 4*1/2 = 2Marks

ARRAY	**LINKED LIST**
Size of an array is fixed	Size of a list is not fixed
Memory is allocated from stack	Memory is allocated from heap
It is necessary to specify the number of elements during declaration (i.e., during compile time).	It is not necessary to specify the number of elements during declaration (i.e., memory is allocated during run time).
It occupies less memory than a linked list for the same number of elements.	It occupies more memory.
Inserting new elements at the front is potentially expensive because existing elements need to be shifted over to make room.	Inserting a new element at any position can be carried out easily.
Deleting an element from an array is not possible.	Deleting an element is possible.

Operations on Linked List: (Any 4) 4*1/4 = 1Mark

1. Inserting a node at the beginning.
2. Inserting a node at the end.
3. Inserting a node at intermediate position.
4. Delete a node at the end.
5. Delete the node at the beginning
6. Display

3. Define Header node. What is the difference between Singly Linked List and Doubly Linked List with respect to the node structure?

Solution:

1 - Mark

A header node is a special dummy node found at the front of the list. The use of header node is an alternative to remove the first node in a list.

Usually header node data field contains the number of nodes present in the list.

1-Mark

Singly Linked list contains one link field, which stores the address of the next node whereas doubly linked list contains 2 link fields, where the left link contains the previous node address and right link contains the address of the next node.

1-Mark

Singly linked list node structure:

Data field	Address field

Doubly linked list node structure:

Address field	Data field	Address field

4. Write a function to delete any duplicate nodes from a singular linked list sorted in non-decreasing order. The list should only be traversed once.

Sol:

Algorithm:

Traverse the list from the head (or start) node. While traversing, compare each node with its next node. If

data of next node is same as current node then delete the next node. Before we delete a node, we need to store next pointer of the node

Implementation:
Functions other than removeDuplicates() are just to create a linked linked list and test removeDuplicates().

```
#include<stdio.h>
#include<stdlib.h>

/* Link list node */
struct Node
{
   int data;
  struct Node* next;
};
```

```
/* The function removes duplicates from a sorted list */
void removeDuplicates(struct Node* head)
{
  /* Pointer to traverse the linked list */
  struct Node* current = head;

  /* Pointer to store the next pointer of a node to be deleted*/
  struct Node* next_next;

  /* do nothing if the list is empty */
  if (current == NULL)
    return;

  /* Traverse the list till last node */
  while (current->next != NULL)
  {
    /* Compare current node with next node */
    if (current->data == current->next->data)
    {
      /* The sequence of steps is important*/
      next_next = current->next->next;
      free(current->next);
      current->next = next_next;
    }
    else /* This is tricky: only advance if no deletion */
    {
      current = current->next;
    }
  }
}
```

- **Declaration 1 mark**
- **List Traversal 2 marks**
- **Node comparision 1 Marks**
- **The Sequence of steps to remove the duplicates 2 Marks**
- **Advance the pointer to next if no deletion required 1 mark**

5. Given a singly linked list, write a function to swap elements pairwise.
Sample Input 1->2->3->4->5 Output : 2->1->4->3->5,

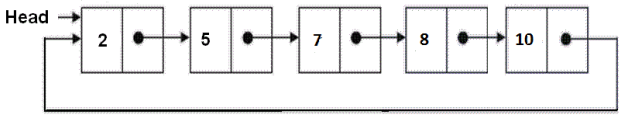
1->2->3->4->5->6 Output 2->1->4->3->6->5.

```
*Function to swap two integers at addresses a and b */
void swap(int *a, int *b);

/* Function to pairwise swap elements of a linked list */
void pairWiseSwap(struct Node *head)
{
   struct Node *temp = head;     1 Mark

   /* Traverse further only if there are at-least two nodes left */
   while (temp != NULL && temp->next != NULL)   -2 marks
   {
      /* Swap data of node with its next node's data */
      swap(&temp->data, &temp->next->data);  2 marks

      /* Move temp by 2 for the next pair */
      temp = temp->next->next;     2 marks
   }
}
```

- **Declaration 1 mark**
- **List traversal 2 marks**
- **Swapping 2 marks**
- **Move pointer by 2 for the next pair 2 marks**

6a. Define Circular linked list? Explain the operations on circular linked list. Write the function to insert and delete a node at a given position.

Sol: A circular linked list is one, which has no beginning and no end. A single linked list can be made a circular linked list by simply storing address of the very first node in the link field of the last node. 1.5m



The basic operations in a circular linked list are: 1.5m

- Creation.
- Insertion.
- Deletion.
- Traversing.

The Function to insert and delete at particular position is given below:

NODE insert_at_pos(NODE head,int pos,int item) 3.5m

```
{
  NODE temp,cur,prev;
  int cn=1;
  temp = (NODE)malloc(sizeof(struct node));
  temp->data = item;
  prev=head;
  cur=head->link;
  while(cur!=head && pos!=cn)
  {
        prev=cur;
        cur=cur->link;
        cn++;
  }
  prev->link=temp;
  temp->link=cur;
  head->data=++count;
return head;
}
```

NODE del_pos(NODE head,int pos) 3.5m

```
{
  NODE cur,prev;
  int cn=1;
  if(pos>head->data)
  {
        printf("deletion not possible Invalid position\n");

  }
  else
  {
  prev=head;
  cur=head->link;
  while(cur!=head && pos!=cn)
  {
        prev=cur;
        cur=cur->link;
        cn++;
  }
        prev->link=cur->link;
  printf("ele deleted is %d\n", cur->data);
  free(cur);
  head->data=--count;
  return head;
  }
}
```

6b. Write an algorithm to implement the function enque() for a QUEUE using aarrys. 5m.

```
#define QSize 5
int front=-1,rear=-1;
Void enque()
{
    int ele;
    printf("Enter the element\n");
    scanf("%d\n",&ele);
    if(front!=rear)
    {
            Q[++rear]=ele;
    }
    else
    printf("Q if FULL\n");
}
```

PRESIDENCY UNIVERSITY, BENGALURU
SCHOOL OF ENGINEERING

Max Marks: 40 Max Time: 60 Mins Weightage: 20 %

TEST 1

I Semester 2017-2018 Course: **CSE 201 Data Structure** 16 SEPT 2017

Instructions:

i. Write legibly
ii. Scientific and non programmable calculators are permitted

---

Part A

(3Q x 3 M= 09 Marks)

1. Define Data structure? What are the different types of data types of structure with example.
2. What is a pointer? Explain with example the syntax and declaration of pointer variable.
3. What is dynamic memory allocation? Name the different types of memory management function

Part B

(7 Q x 3 M= 21 Marks)

1. Define structure and union with syntax. Explain with simple example.
2. Explain how algorithms are analyzed? Differentiate between the recursive and iterative algorithm.
3. Give the recursive algorithm to find the GCD of a two numbers.
4. Define stack? Explain on what principal the stack works with example.
5. What are the operations on stack? Give the algorithm for the operations on stack.
6. What are the applications of stack?
7. Convert the following infix expression to postfix and evaluate the postfix expression.
   i) ((2+(4-3)*6)^1+4)
   ii) (7-2*(3+4)-3*2)*5

Part C

(1 Q x 10 M= 10Marks)

1. What is postfix expression? Write the algorithm to convert the infix to postfix expression .Convert the given infix expression to postfix using the algorithm.
   A * (B + C * D) + E