Jadavpur University
Department of Computer Science & Engg
MCA, 1st Year, 2nd Semester
Data and File Structures, Class Test-I

Max Time: 1 hour **Max Marks: 30**

Marks distribution: for each question, 1 for correct the choice and 2 for an explanation of such choice.

1. In list implementation, a node carries information regarding
   a) The data
   b) The link
   c) The link and the data
   d) Addresses of the elements
   e) None of the above
2. Linked list data structure usage offers considerable saving in
   a) Computational time
   b) Space utilization
   c) Space utilization as well as computation time.
   d) None of the above
3. Which of the following operations is performed more efficiently by double linked list than by linear linked list?
   a) Deletion a node whose location is given
   b) Searching an unsorted list for a given item
   c) Inserting a node after the node with a given location
   d) None of these
4. Which of the following statement(s) is/are true regarding insertion of node in a linear linked list?
   a) Setting the field of the new node means allocating memory to newly created node
   b) If the node precedes all others in the list, then insert it at the front and return its address
   c) Creating a new node depends upon free memory space
   d) All of these
   e) None of these
5. Which of the following operations is performed more efficiently by double linked list than by linear linked list?
   a) Deletion a node whose location is given
   b) Searching an unsorted list for a given item
   c) Inserting a node after the node with a given location
   d) None of these
6. Convert the following Infix expression to Postfix form
   x + y * z + (p * q + r) * s, Follow usual precedence rule and assume that the expression is legal.
   a) xyz*+pq*r+s*+
   b) xyz*+pq*r+s+*
   c) xyz+*pq*r+s*+
   d) None of these
7. What does the following function do for a given Linked List with first node as head?

```
void fun1(struct node* head)
{
 if(head == NULL)
  return;

fun1(head->next);
printf("%d  ", head->data);
}
```

   a) Prints all nodes of linked lists
   b) Prints all nodes of linked list in reverse order
   c) Prints alternate nodes of Linked List
   d) Prints alternate nodes in reverse order
   e) None of the Above
8. What is the output of following function for start pointing to first node of following linked list?
   1->2->3->4->5->6

```
void fun(struct node* start)
```

```
{
 if(start == NULL)
   return;
 printf("%d  ", start->data);

 if(start->next != NULL )
   fun(start->next->next);
 printf("%d  ", start->data);
}
```

a) 1 4 6 6 4 1
b) 1 3 5 1 3 5
c) 1 2 3 5
d) 1 3 5 5 3 1
e) None of the above

9. The following C function takes a simply-linked list as input argument. It modifies the list by moving the last element to the front of the list and returns the modified list. Some part of the code is left blank. Choose the correct alternative to replace the blank line.

```
typedef struct node
{
 int value;
 struct node *next;
}Node;

Node *move_to_front(Node *head)
{
 Node *p, *q;
 if ((head == NULL: || (head->next == NULL))
   return head;
 q = NULL; p = head;
 while (p-> next !=NULL)
 {
   q = p;
   p = p->next;
 }
 _______________________________
 return head;
}
```

a) q = NULL; p->next = head; head = p;
b) q->next = NULL; head = p; p->next = head;
c) head = p; p->next = q; q->next = NULL;
d) q->next = NULL; p->next = head; head = p;
e) None of the above

10. Consider the following function to traverse a linked list.

```
void traverse(struct Node *head)
{
while (head->next != NULL)
{
printf("%d  ", head->data);
head = head->next;
}
}
```

Which of the following is **FALSE** about above function?

a) The function may crash when the linked list is empty
b) The function doesn't print the last node when the linked list is not empty
c) The function is implemented incorrectly because it changes head
d) Both (a) and (b)
e) All; (a), (b) and (c)
f) None of the above