JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT

TEST -1 EXAMINATION- 2024

M.Tech.-I Semester (CSE-IS)

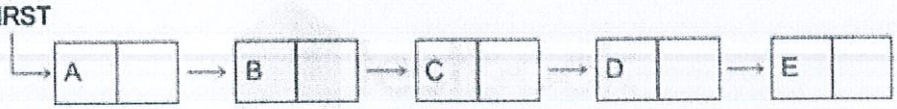
COURSE CODE (CREDITS): 10M11CI111 MAX. MARKS: 15

COURSE NAME: Advanced Data Structures

COURSE INSTRUCTORS: Ekta Gandotra MAX. TIME: 1 Hr.

Note: *(a)All questions are compulsory.*
(b)Marks are indicated against each question in square brackets.
(c) The candidate is allowed to make Suitable numeric assumptions wherever required for solving problems.

Q1. [3] CO1



Find the output if the following sequence of operations are performed on the given singly linked list.

struct node *P
i) P=First -> link -> link;
ii) P->link ->link -> link = First;
iii) P= P -> link -> link;
iv) printf("%c", P->data);

Q2. a. Let P be a singly linked list and Q be the pointer to an intermediate node X in the list. What is the worst-case time complexity of the best-known algorithm to delete node X from the list? Justify your answer. [2] CO1

b. Write a pseudocode to insert a new element in the beginning of the circular linked list in O(1) time complexity. [2]

Q3. Consider the following sequence of operations on an empty stack: push(54); push(52); pop(); push(55); push(62); A = pop(); Consider the following sequence of operations on an empty queue: enqueue(21); enqueue(24); dequeue(); enqueue(28); enqueue(32); B = dequeue(); Find the value of A + B? Show all steps of your calculation. [2] CO1

Q4. a. Write the overflow condition for circular queue. [1] CO1

b. Analyze the time complexity of insertion, deletion, and search operations in a priority queue implemented using a singly linked list. [2]

Q5. Consider a doubly linked list with every node having data part, previous pointer (as prev) and next pointer (as next). Following function takes reference to the head of the doubly linked list as an argument. What would be the modified doubly linked list after the function call if reference to the head of 1 <--> 2 <--> 3 <--> 4 <--> 5 <-->6 linked list passed as a parameter to the function? Justify your answer. [3] CO1

```
void fun(struct node **head_ref)
{
    struct node *temp = NULL;
    struct node *current = *head_ref;

    while (current !=  NULL)
    {
        temp = current->prev;
        current->prev = current->next;
        current->next = temp;
        current = current->prev;
    }

    if(temp != NULL )
        *head_ref = temp->prev;
}
```