Khwopa College of Engineering

DSA Lab Sheet

II Year/ II Part

Faculty: Computer and Electronics

Labsheet #5

Objectives:

1. Stack & Queue using Linked List and Polynomial Addition using Linked List.

Theory:

1. Stack using Linked List

A stack is a Last In First Out (LIFO) data structure.

Operations:

- push(): Insert element at the top.
- pop(): Remove element from the top.
- peek(): View top element.

Example Code:

```
#include <stdio.h>
#include <stdlib.h>
// Define structure for a node
struct Node {
    int data;
    struct Node* next;
};
// Global top pointer
struct Node* top = NULL;
// Push operation
void push(int value) {
    struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
    if (newNode == NULL) {
        printf("Stack Overflow\n");
        return;
    }
    newNode->data = value;
    newNode->next = top;
    top = newNode;
    printf("%d pushed to stack.\n", value);
}
// Pop operation
void pop() {
    if (top == NULL) {
        printf("Stack Underflow\n");
        return;
    }
    struct Node* temp = top;
    printf("%d popped from stack.\n", top->data);
    top = top->next;
    free(temp);
}
```

```
// Peek operation
void peek() {
    if (top == NULL)
        printf("Stack is empty.\n");
    else
        printf("Top element: %d\n", top->data);
}
// Display operation
void display() {
    struct Node* temp = top;
    if (top == NULL) {
        printf("Stack is empty.\n");
        return;
    }
    printf("Stack elements: ");
    while (temp != NULL) {
        printf("%d ", temp->data);
        temp = temp->next;
    }
    printf("\n");
}

// Main function with sample usage
int main() {
    push(10);
    push(20);
    push(30);
    display();      // Output: 30 20 10
    pop();          // Output: 30 popped
    peek();         // Output: 20
    display();      // Output: 20 10
    return 0;
}
```

2 . Queue using Linked List

A **queue** is a First In First Out (FIFO) data structure.

Operations:

- enqueue(): Insert at rear
- dequeue(): Remove from front
- display(): Show all elements

Functions to Implement:

```
void enqueue(int data);

void dequeue();

void display();
```

3 . Polynomial Addition using Linked List

Polynomials like 4x^3 + 2x^2 + 5 are stored as linked list nodes.

Each node contains:

- Coefficient
- Exponent
- Addition involves matching exponents and adding coefficients.

Structure Definition:

```
struct Node {
     int coeff;
      int exp;
     struct Node* next;
};
```

Sample Task :

- Input two polynomials:
    - Poly1: `3x^3 + 2x^2 + 1`
    - Poly2: `5x^2 + 2x + 1`
- Output: `3x^3 + 7x^2 + 2x + 2`

Functions to Implement :

```
struct Node* createPoly();
struct Node* addPoly(struct Node* poly1, struct Node* poly2);
void displayPoly(struct Node* poly);
```