4. WAP to Implement Singly Linked List with following operations a) Create a linked list. b) Insertion of a node at first position, at any position and at end of list. Display the contents of the linked list.

```
#include <stdio.h>

#include <stdlib.h>


// Define a node structure

struct Node {

    int data;

    struct Node* next;

};


// Function to create a new node

struct Node* createNode(int value) {

    struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));

    newNode->data = value;

    newNode->next = NULL;

    return newNode;

}


// Function to insert at the first position

void insertFirst(struct Node** head, int value) {

    struct Node* newNode = createNode(value);

    newNode->next = *head;

    *head = newNode;

}


// Function to insert at any position

void insertAtPosition(struct Node** head, int value, int position) {

    struct Node* newNode = createNode(value);

    struct Node* temp = *head;


    // If inserting at position 1 (first position)
```

```
    if (position == 1) {
        newNode->next = *head;
        *head = newNode;
        return;
    }

    // Traverse the list to the desired position
    for (int i = 1; temp != NULL && i < position - 1; i++) {
        temp = temp->next;
    }

    // If position is greater than the number of nodes, do nothing
    if (temp == NULL) {
        printf("Position is greater than the length of the list.\n");
        return;
    }

    // Insert the new node at the desired position
    newNode->next = temp->next;
    temp->next = newNode;
}

// Function to insert at the end of the list
void insertEnd(struct Node** head, int value) {
    struct Node* newNode = createNode(value);

    // If the list is empty, make the new node the first node
    if (*head == NULL) {
        *head = newNode;
        return;
    }
```

```
    // Traverse to the last node
    struct Node* temp = *head;
    while (temp->next != NULL) {
        temp = temp->next;
    }

    // Insert the new node at the end
    temp->next = newNode;
}

// Function to display the contents of the linked list
void displayList(struct Node* head) {
    if (head == NULL) {
        printf("The list is empty.\n");
        return;
    }

    struct Node* temp = head;
    printf("Linked List: ");
    while (temp != NULL) {
        printf("%d ", temp->data);
        temp = temp->next;
    }
    printf("\n");
}

// Main function
int main() {
    struct Node* head = NULL;
    int choice, value, position;

    while (1) {
```

```
// Menu for the operations
printf("\nLinked List Operations Menu:\n");
printf("1. Create Linked List\n");
printf("2. Insert at First Position\n");
printf("3. Insert at Any Position\n");
printf("4. Insert at End\n");
printf("5. Display Linked List\n");
printf("6. Exit\n");
printf("Enter your choice: ");
scanf("%d", &choice);

switch (choice) {
    case 1:
        printf("Enter value to create the linked list: ");
        scanf("%d", &value);
        head = createNode(value);
        printf("Linked list created with value %d.\n", value);
        break;

    case 2:
        printf("Enter value to insert at first position: ");
        scanf("%d", &value);
        insertFirst(&head, value);
        printf("Inserted %d at first position.\n", value);
        break;

    case 3:
        printf("Enter value and position to insert: ");
        scanf("%d %d", &value, &position);
        insertAtPosition(&head, value, position);
        printf("Inserted %d at position %d.\n", value, position);
        break;
```

```
            case 4:
                printf("Enter value to insert at end: ");
                scanf("%d", &value);
                insertEnd(&head, value);
                printf("Inserted %d at end.\n", value);
                break;

            case 5:
                displayList(head);
                break;

            case 6:
                printf("Exiting...\n");
                return 0;

            default:
                printf("Invalid choice! Please try again.\n");
        }
    }

    return 0;
}
```

```
Linked List Operations Menu:
1. Create Linked List
2. Insert at First Position
3. Insert at Any Position
4. Insert at End
5. Display Linked List
6. Exit
Enter your choice: 1
Enter value to create the linked list: 10
Linked list created with value 10.

Linked List Operations Menu:
1. Create Linked List
2. Insert at First Position
3. Insert at Any Position
4. Insert at End
5. Display Linked List
6. Exit
Enter your choice: 2
Enter value to insert at first position: 20
Inserted 20 at first position.

Linked List Operations Menu:
1. Create Linked List
2. Insert at First Position
3. Insert at Any Position
4. Insert at End
5. Display Linked List
6. Exit
Enter your choice: 3
Enter value and position to insert: 30
4
Position is greater than the length of the list.
Inserted 30 at position 4.

Linked List Operations Menu:
1. Create Linked List
2. Insert at First Position
3. Insert at Any Position
4. Insert at End
5. Display Linked List
6. Exit
Enter your choice: 4
Enter value to insert at end: 40
Inserted 40 at end.

Linked List Operations Menu:
1. Create Linked List
2. Insert at First Position
3. Insert at Any Position
4. Insert at End
5. Display Linked List
6. Exit
Enter your choice: 5
Linked List: 20 10 40
```