Program 4

Write A Program 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.
c) Display the contents of the linked list.

Code:

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

struct Node {
   int data;
   struct Node* next;
};

void createList(struct Node** head);
void insertAtBeginning(struct Node** head, int data);
void insertAtPosition(struct Node** head, int data, int position);
void insertAtEnd(struct Node** head, int data);
void displayList(struct Node* head);

int main() {
   struct Node* head = NULL;
   int choice, data, position;

   while (1) {
      printf("\nMenu:\n");
      printf("1. Create a linked list\n");
      printf("2. Insert at the beginning\n");
      printf("3. Insert at a specific position\n");
      printf("4. Insert at the end\n");
      printf("5. Display the list\n");
      printf("6. Exit\n");
      printf("Enter your choice: ");
      scanf("%d", &choice);

      switch (choice) {
         case 1:
            createList(&head);
            break;
         case 2:
            printf("Enter data to insert at the beginning: ");
            scanf("%d", &data);
            insertAtBeginning(&head, data);
            break;
         case 3:
            printf("Enter data to insert: ");
            scanf("%d", &data);
```

```
            printf("Enter position to insert (starting from 1): ");
            scanf("%d", &position);
            insertAtPosition(&head, data, position);
            break;
        case 4:
            printf("Enter data to insert at the end: ");
            scanf("%d", &data);
            insertAtEnd(&head, data);
            break;
        case 5:
            displayList(head);
            break;
        case 6:
            printf("Exiting the program.\n");
            exit(0);
        default:
            printf("Invalid choice. Please try again.\n");
      }
   }

   return 0;
}

void createList(struct Node** head) {
   int data, choice;
   do {
      printf("Enter data to insert: ");
      scanf("%d", &data);
      insertAtEnd(head, data);
      printf("Do you want to add another node? (1 for Yes, 0 for No): ");
      scanf("%d", &choice);
   } while (choice != 0);
}

void insertAtBeginning(struct Node** head, int data) {
   struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
   if (newNode == NULL) {
      printf("Memory allocation failed.\n");
      return;
   }
   newNode->data = data;
   newNode->next = *head;
   *head = newNode;
   printf("Node inserted at the beginning.\n");
}

void insertAtPosition(struct Node** head, int data, int position) {
   if (position < 1) {
      printf("Invalid position.\n");
      return;
```

```
    }

    struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
    if (newNode == NULL) {
        printf("Memory allocation failed.\n");
        return;
    }
    newNode->data = data;

    if (position == 1) {
        newNode->next = *head;
        *head = newNode;
        printf("Node inserted at position %d.\n", position);
        return;
    }

    struct Node* temp = *head;
    for (int i = 1; i < position - 1 && temp != NULL; i++) {
        temp = temp->next;
    }

    if (temp == NULL) {
        printf("Position out of bounds.\n");
        free(newNode);
        return;
    }

    newNode->next = temp->next;
    temp->next = newNode;
    printf("Node inserted at position %d.\n", position);
}

void insertAtEnd(struct Node** head, int data) {
    struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
    if (newNode == NULL) {
        printf("Memory allocation failed.\n");
        return;
    }
    newNode->data = data;
    newNode->next = NULL;

    if (*head == NULL) {
        *head = newNode;
    } else {
        struct Node* temp = *head;
        while (temp->next != NULL) {
            temp = temp->next;
        }
        temp->next = newNode;
    }
```

```
    printf("Node inserted at the end.\n");
}

void displayList(struct Node* head) {
    if (head == NULL) {
        printf("The list is empty.\n");
        return;
    }

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

```
Menu:
1. Create a linked list
2. Insert at the beginning
3. Insert at a specific position
4. Insert at the end
5. Display the list
6. Exit
Enter your choice: 2
Enter data to insert at the beginning: 12
Node inserted at the beginning.

Menu:
1. Create a linked list
2. Insert at the beginning
3. Insert at a specific position
4. Insert at the end
5. Display the list
6. Exit
Enter your choice: 2
Enter data to insert at the beginning: 23
Node inserted at the beginning.

Menu:
1. Create a linked list
2. Insert at the beginning
3. Insert at a specific position
4. Insert at the end
5. Display the list
6. Exit
Enter your choice: 3
Enter data to insert: 2
Enter position to insert (starting from 1): 2
Node inserted at position 2.
```

```
Menu:
1. Create a linked list
2. Insert at the beginning
3. Insert at a specific position
4. Insert at the end
5. Display the list
6. Exit
Enter your choice: 5
Linked list contents: 23 -> 2 -> 12 -> NULL

Menu:
1. Create a linked list
2. Insert at the beginning
3. Insert at a specific position
4. Insert at the end
5. Display the list
6. Exit
Enter your choice: 6
Exiting the program.
```

28/10/24

Q5 a) WAP to Implement Singly Linked List by following operations
→ Create a linked list
→ insertion of a node at first position and at end of the list
display the contents of linked lists

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

struct Node {
    int data;
    struct Node * next;
};

struct Node * createNode(int data) {
    struct Node * newNode = (struct Node *) malloc
                (sizeof (struct Node));
    newNode -> data = data;
    newNode -> next = NULL;
    return newNode;
}

void insertAtBeginning(struct Node ** head, int data) {
    struct Node * newNode = createNode(data);
    newNode -> next = * head;
    * head = newNode;
}

void insertAtPosition(struct Node ** head, int data,
                      int pos) {
    if (pos == 1) {
    newNode -> next = * head;
    * head = newNode;
    return;
}
    struct Node * temp = * head;
    for(int i = 1; i < pos-1 && temp != NULL; i++) {
        temp = temp -> next;
    }
    if (temp == NULL) {
        printf("Position out of range\n");
    }
    else {
        newNode -> next = temp -> next;
        temp -> next = newNode;
    }
}

void insertAtEnd(struct Node ** head,
                 int data) {
    struct Node * newNode = createNode
                (data);
    if (* head == NULL) {
        * head = newNode;
        return;
    }
    struct Node * temp = * head
    while (temp -> next != NULL) {
            temp = temp -> next;
    }
    temp -> next = newNode;
}

void display(struct Node * head) {
    if (head == NULL) {
        printf("Empty list\n");
        return;
    }
    struct Node * temp = head;
    while (temp != NULL) {
        printf("%d -> ", temp -> data);
        temp = temp -> next;
    }
    printf("NULL\n");
}

int main() {
    struct Node * head = NULL

    insertAtBeginning(&head, 5);
    display(head);

    insertAtPos(&head, 15, 3);
    display(head);

    insertAtEnd(&head, 30);
    display(head);

    return 0;
}
```

output

1. Insert at the beginning
2. Insert at the end
3. Display the list
4. Exit

Enter your choice : 1
Enter value to insert at the beginning : 12

Enter your choice : 2
Enter the value to insert at the end : 23

Enter your choice : 3
linked list : 12 → 23 → NULL

Enter your choice : 1
enter : 98

Enter your choice : 2
enter : 568

Enter your choice 3
enter :
linked list : 98 → 12 → 23 → 568
→ NULL