Name: Vinayak Kumar Singh **Subject:** DSA Lab

Register No: 23MCA1030 **Date**:05-09-2023

Exercise 7: Double Linked List

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

struct Node {
    int data;
    struct Node* prev;
    struct Node* next;
};
struct Node* head = NULL;

void insertAtBeginning(int value) {
    struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
    if (newNode == NULL) {
        printf("No Memory to store data\n");
        return;
    }
    newNode->data = value;
    newNode->prev = NULL;
    newNode->next = head;

    if (head != NULL)
        head->prev = newNode;

    head = newNode;
    printf("Data %d is Inserted at the beginning\n", value);
}

void insertAtEnd(int value) {
    struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
    if (newNode == NULL) {
        printf("No Memory to store data\n");
        return;
    }
    newNode->data = value;
    newNode->next = NULL;

    if (head == NULL) {
        newNode->prev = NULL;
        head = newNode;
        printf("Data %d is Inserted at the end.\n", value);
        return;
```

```
    }

    struct Node* temp = head;
    while (temp->next != NULL)
        temp = temp->next;

    temp->next = newNode;
    newNode->prev = temp;
    printf("Data %d is Inserted at the end\n", value);
}

void insertAtAfter(int afterValue, int value) {
    struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
    if (newNode == NULL) {
        printf("No Memory to store data\n");
        return;
    }
    newNode->data = value;

    struct Node* temp = head;
    while (temp != NULL && temp->data != afterValue)
        temp = temp->next;

    if (temp == NULL) {
        printf("Data %d is not found in the list\n", afterValue);
        free(newNode);
        return;
    }

    newNode->next = temp->next;
    if (temp->next != NULL)
        temp->next->prev = newNode;

    newNode->prev = temp;
    temp->next = newNode;

    printf("Inserted %d after %d\n", value, afterValue);
}

void deleteBeginning() {
    if (head == NULL) {
        printf("List is empty\n");
        return;
    }

    struct Node* temp = head;
    head = head->next;
    if (head != NULL)
        head->prev = NULL;
```

```
    free(temp);
    printf("Deleted the first element.\n");
}

void deleteEnd() {
    if (head == NULL) {
        printf("List is empty\n");
        return;
    }

    if (head->next == NULL) {
        free(head);
        head = NULL;
        printf("Deleted the last element\n");
        return;
    }

    struct Node* temp = head;
    while (temp->next != NULL)
        temp = temp->next;

    temp->prev->next = NULL;
    free(temp);
    printf("Deleted the last element\n");
}

void deleteSpecific(int value) {
    if (head == NULL) {
        printf("List is empty.\n");
        return;
    }

    struct Node* temp = head;
    while (temp != NULL && temp->data != value)
        temp = temp->next;

    if (temp == NULL) {
        printf("Data %d not found in the list.\n", value);
        return;
    }

    if (temp->prev != NULL)
        temp->prev->next = temp->next;
    else
        head = temp->next;

    if (temp->next != NULL)
        temp->next->prev = temp->prev;

    free(temp);
```

```
    printf("Deleted %d from the list\n", value);
}

void display() {
    if (head == NULL) {
        printf("List is empty\n");
        return;
    }

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

int main() {
    int mainmenu, choice, value, afterValue;

    while (1) {
        printf("\nChoose from the Doubly Linked List Menu:\n");
        printf("1.Insert\n");
        printf("2.Delete\n");
        printf("3.Display\n");
        printf("4.Exit\n");

        printf("Enter your choice : ");
        scanf("%d", &mainmenu);

        switch (mainmenu) {
            case 1:
                printf("Choose an insertion option:\n");
                printf("1. Insert at Beginning\n");
                printf("2. Insert at End\n");
                printf("3. Insert after a Value\n");
                printf("Enter your choice: ");
                scanf("%d", &choice);

                switch (choice) {
                    case 1:
                        printf("Enter the value to insert at the beginning: ");
                        scanf("%d", &value);
                        insertAtBeginning(value);
                        break;
                    case 2:
                        printf("Enter the value to insert at the end: ");
                        scanf("%d", &value);
                        insertAtEnd(value);
```

```
                    break;
                case 3:
                    printf("Enter the value to insert: ");
                    scanf("%d", &value);
                    printf("Enter the value after which to insert: ");
                    scanf("%d", &afterValue);
                    insertAtAfter(afterValue, value);
                    break;
                default:
                    printf("Invalid choice for insertion.\n");
            }
            break;
        case 2:
            printf("Choose a deletion option:\n");
            printf("1. Delete from Beginning\n");
            printf("2. Delete from End\n");
            printf("3. Delete by Value\n");
            printf("Enter your choice: ");
            scanf("%d", &choice);

            switch (choice) {
                case 1:
                    deleteBeginning();
                    break;
                case 2:
                    deleteEnd();
                    break;
                case 3:
                    printf("Enter the value to delete: ");
                    scanf("%d", &value);
                    deleteSpecific(value);
                    break;
                default:
                    printf("Invalid choice for deletion.\n");
            }
            break;
        case 3:
            display();
            break;
        case 4:
            exit(0);
        default:
            printf("Please choose from available options\n");
        }
    }

    return 0;
}
```

Output:

```
Choose from the Doubly Linked List Menu:
1.Insert
2.Delete
3.Display
4.Exit
Enter your choice : 1
Choose an insertion option:
1. Insert at Beginning
2. Insert at End
3. Insert after a Value
Enter your choice: 1
Enter the value to insert at the beginning: 10
Data 10 is Inserted at the beginning

Choose from the Doubly Linked List Menu:
1.Insert
2.Delete
3.Display
4.Exit
Enter your choice : 1
Choose an insertion option:
1. Insert at Beginning
2. Insert at End
3. Insert after a Value
Enter your choice: 2
Enter the value to insert at the end: 20
Data 20 is Inserted at the end
```

```
Choose from the Doubly Linked List Menu:
1.Insert
2.Delete
3.Display
4.Exit
Enter your choice : 1
Choose an insertion option:
1. Insert at Beginning
2. Insert at End
3. Insert after a Value
Enter your choice: 3
Enter the value to insert: 30
Enter the value after which to insert: 20
Inserted 30 after 20

Choose from the Doubly Linked List Menu:
1.Insert
2.Delete
3.Display
4.Exit
Enter your choice : 3
Doubly Linked List: 10 20 30

Choose from the Doubly Linked List Menu:
1.Insert
2.Delete
3.Display
4.Exit
Enter your choice : 2
Choose a deletion option:
1. Delete from Beginning
2. Delete from End
3. Delete by Value
Enter your choice: 1
Deleted the first element.
```

```
Choose from the Doubly Linked List Menu:
1.Insert
2.Delete
3.Display
4.Exit
Enter your choice : 2
Choose a deletion option:
1. Delete from Beginning
2. Delete from End
3. Delete by Value
Enter your choice: 2
Deleted the last element

Choose from the Doubly Linked List Menu:
1.Insert
2.Delete
3.Display
4.Exit
Enter your choice : 1
Choose an insertion option:
1. Insert at Beginning
2. Insert at End
3. Insert after a Value
Enter your choice: 1
Enter the value to insert at the beginning: 40
Data 40 is Inserted at the beginning

Choose from the Doubly Linked List Menu:
1.Insert
2.Delete
3.Display
4.Exit
Enter your choice : 3
Doubly Linked List: 40 20
```

```
Choose from the Doubly Linked List Menu:
1.Insert
2.Delete
3.Display
4.Exit
Enter your choice : 2
Choose a deletion option:
1. Delete from Beginning
2. Delete from End
3. Delete by Value
Enter your choice: 3
Enter the value to delete: 20
Deleted 20 from the list

Choose from the Doubly Linked List Menu:
1.Insert
2.Delete
3.Display
4.Exit
Enter your choice : 3
Doubly Linked List: 40
```

```
Choose from the Doubly Linked List Menu:
1.Insert
2.Delete
3.Display
4.Exit
Enter your choice : 4
```

END