DSA ASSIGNMENT III

Singly and Doubly Linked List

By Naveen Jayaraj

RA2311026050064

ANSWERS

1)

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

struct Node {
    char name[50];
    struct Node* next;
};

struct Node* createNode(char* name) {
    struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
    strcpy(newNode->name, name);
    newNode->next = NULL;
    return newNode;
}

void displayList(struct Node* head) {
    struct Node* temp = head;
    if (temp == NULL) {
        printf("Contact list is empty.\n");
        return;
    }
    printf("Contact list: ");
    while (temp != NULL) {
        printf("%s -> ", temp->name);
        temp = temp->next;
    }
    printf("NULL\n");
}

struct Node* createList() {
    int n;
    char name[50];
    struct Node* head = NULL;
    struct Node* temp = NULL;

    printf("Enter the number of contacts: ");
    scanf("%d", &n);
    getchar();

    for (int i = 0; i < n; i++) {
        printf("Enter contact name %d: ", i + 1);
        fgets(name, 50, stdin);
        name[strcspn(name, "\n")] = 0;

        struct Node* newNode = createNode(name);

        if (head == NULL) {
            head = newNode;
        } else {
            temp->next = newNode;
        }
        temp = newNode;
    }
    displayList(head);
    return head;
}

struct Node* insertContact(struct Node* head) {
    char name[50];
    int pos, i = 0;
    printf("Enter the contact's name to insert: ");
    getchar();
    fgets(name, 50, stdin);
    name[strcspn(name, "\n")] = 0;

    printf("Enter the position (0-based index) to insert the contact: ");
    scanf("%d", &pos);

    struct Node* newNode = createNode(name);

    if (pos == 0) {
        newNode->next = head;
        head = newNode;
    } else {
        struct Node* temp = head;
        while (i < pos - 1 && temp != NULL) {
            temp = temp->next;
            i++;
        }

        if (temp == NULL) {
```

```
    displayList(head);
    return head;
}

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

    char choice, name[50];
    int pos;

    printf("Delete by name or position? (n/p): ");
    getchar();
    scanf("%c", &choice);

    if (choice == 'n') {
        printf("Enter the contact's name to delete: ");
        getchar();
        fgets(name, 50, stdin);
        name[strcspn(name, "\n")] = 0;

        struct Node* temp = head;
        struct Node* prev = NULL;

        while (temp != NULL && strcmp(temp->name, name) != 0) {
            prev = temp;
            temp = temp->next;
        }

        if (temp == NULL) {
            printf("Contact not found.\n");
            return head;
        }

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

        free(temp);
    } else if (choice == 'p') {
        printf("Enter the position to delete the contact: ");
        scanf("%d", &pos);

        if (pos == 0) {
            struct Node* temp = head;
            head = head->next;
            free(temp);
        } else {
            struct Node* temp = head;
            struct Node* prev = NULL;
            int i = 0;

            while (i < pos && temp != NULL) {
                prev = temp;
                temp = temp->next;
                i++;
            }

            if (temp == NULL) {
                printf("Invalid position.\n");
                return head;
            }

            prev->next = temp->next;
            free(temp);
        }
    } else {
        printf("Invalid choice.\n");
    }

    displayList(head);
    return head;
}

void searchContact(struct Node* head) {
    char name[50];
    int pos = 0;
    struct Node* temp = head;

    printf("Enter the contact's name to search: ");
    getchar();
    fgets(name, 50, stdin);
    name[strcspn(name, "\n")] = 0;
```

```c
    while (temp != NULL) {
        if (strcmp(temp->name, name) == 0) {
            printf("%s found at position %d\n", name, pos);
            return;
        }
        temp = temp->next;
        pos++;
    }

    printf("%s not found in the list.\n", name);
}

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

    do {
        printf("\n1. Create the list of contacts\n");
        printf("2. Insert a new contact\n");
        printf("3. Delete a contact\n");
        printf("4. Display contact list\n");
        printf("5. Search for a contact\n");
        printf("6. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                head = createList();
                break;
            case 2:
                head = insertContact(head);
                break;
            case 3:
                head = deleteContact(head);
                break;
            case 4:
                displayList(head);
                break;
            case 5:
                searchContact(head);
                break;
            case 6:
                printf("Exiting the program...\n");
                break;
            default:
                printf("Invalid choice. Please try again.\n");
        }
    } while (choice != 6);

    return 0;
}
```

Output:

```
1. Create the list of contacts
2. Insert a new contact
3. Delete a contact
4. Display contact list
5. Search for a contact
6. Exit
Enter your choice:
```

```
Enter your choice: 1
Enter the number of contacts: 3
Enter contact name 1: Manu
Enter contact name 2: NaveenA
Enter contact name 3: Aswin
Contact list: Manu -> Naveen -> Aswin -> NULL
```

```
Enter your choice: 2
Enter the contact's name to insert: Madhav
Enter the position (0-based index) to insert the contact: 3
Contact list: Manu -> Naveen -> Aswin -> Madhav -> NULL
```

```
Enter your choice: 3
Delete by name or position? (n/p): p
Enter the position to delete the contact: 3
Contact list: Manu -> Naveen -> Aswin -> NULL
```

```
Enter your choice: 4
Contact list: Manu -> Naveen -> Aswin -> NULL
```

```
Enter your choice: 5
Enter the contact's name to search: Manu
Manu found at position 0
```

```
Enter your choice: 6
Exiting the program...
```

2)

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

struct Node {
    char name[50];
    struct Node* prev;
    struct Node* next;
};

int main() {
    struct Node *head = NULL, *tail = NULL, *temp, *newNode;
    int choice, pos, size = 0, i;
    char name[50], delChoice;

    do {
        printf("\n1. Create the list of contacts\n");
        printf("2. Insert a new contact\n");
        printf("3. Delete a contact\n");
        printf("4. Display contact list\n");
        printf("5. Search for a contact\n");
        printf("6. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);
```

```c
        switch (choice) {
            case 1:
                printf("Enter the number of contacts: ");
                scanf("%d", &size);
                getchar();

                for (i = 0; i < size; i++) {
                    printf("Enter contact name %d: ", i + 1);
                    fgets(name, 50, stdin);
                    name[strcspn(name, "\n")] = 0;

                    newNode = (struct Node*)malloc(sizeof(struct Node));
                    strcpy(newNode->name, name);
                    newNode->prev = NULL;
                    newNode->next = NULL;

                    if (head == NULL) {
                        head = newNode;
                        tail = newNode;
                    } else {
                        tail->next = newNode;
                        newNode->prev = tail;
                        tail = newNode;
                    }
                }
                break;
```

```c
case 2:
    printf("Enter the contact's name to insert: ");
    getchar();
    fgets(name, 50, stdin);
    name[strcspn(name, "\n")] = 0;

    printf("Enter the position (0-based index) to insert the contact: ");
    scanf("%d", &pos);
    if (pos < 0 || pos > size) {
        printf("Invalid position.\n");
        break;
    }
    newNode = (struct Node*)malloc(sizeof(struct Node));
    strcpy(newNode->name, name);
    newNode->prev = NULL;
    newNode->next = NULL;

    if (pos == 0) {
        newNode->next = head;
        if (head != NULL) {
            head->prev = newNode;  }
        head = newNode;
    } else {
        temp = head;
        for (i = 0; i < pos - 1; i++) {
            temp = temp->next;
        }
        newNode->next = temp->next;
        if (temp->next != NULL) {
            temp->next->prev = newNode;
        }
        temp->next = newNode;
        newNode->prev = temp;
    }
    size++;
    break;
```

```c
case 3:
    printf("Delete by name or position? (n/p): ");
    getchar();
    scanf("%c", &delChoice);

    if (delChoice == 'n') {
        printf("Enter the contact's name to delete: ");
        getchar();
        fgets(name, 50, stdin);
        name[strcspn(name, "\n")] = 0;

        temp = head;
        while (temp != NULL && strcmp(temp->name, name) != 0) {
            temp = temp->next;
        }

        if (temp == NULL) {
            printf("Contact not found.\n");
            break;
        }
    } else if (delChoice == 'p') {
        printf("Enter the position (0-based index) to delete the contact: ");
        scanf("%d", &pos);

        if (pos < 0 || pos >= size) {
            printf("Invalid position.\n");
            break;
        }

        temp = head;
        for (i = 0; i < pos; i++) {
            temp = temp->next;
        }
    } else {
        printf("Invalid choice.\n");
        break;
    }
```

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

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

            free(temp);
            size--;
            break;

        case 4:
            temp = head;
            printf("Contact list (forward): ");
            while (temp != NULL) {
                printf("%s <-> ", temp->name);
                tail = temp;
                temp = temp->next;
            }
            printf("NULL\n");

            temp = tail;
            printf("Contact list (backward): ");
            while (temp != NULL) {
                printf("%s <-> ", temp->name);
                temp = temp->prev;
            }
            printf("NULL\n");
            break;
```

```
        case 5:
            printf("Enter the contact's name to search: ");
            getchar();
            fgets(name, 50, stdin);
            name[strcspn(name, "\n")] = 0;

            temp = head;
            i = 0;
            while (temp != NULL) {
                if (strcmp(temp->name, name) == 0) {
                    printf("%s found at position %d\n", name, i);
                    break;
                }
                temp = temp->next;
                i++;
            }

            if (temp == NULL) {
                printf("%s not found in the list.\n", name);
            }
            break;

        case 6:
            printf("Exiting the program...\n");
            break;

        default:
            printf("Invalid choice. Please try again.\n");
    }
} while (choice != 6);

return 0;
}
```

Output:

```
Output
/tmp/dEEBYOcwMK.o

1. Create the list of contacts
2. Insert a new contact
3. Delete a contact
4. Display contact list
5. Search for a contact
6. Exit
```

```
Enter your choice: 1
Enter the number of contacts: 3
Enter contact name 1: Naveen
Enter contact name 2: Manu
Enter contact name 3: Sujin
```

```
Enter your choice: 2
Enter the contact's name to insert: Livins
Enter the position (0-based index) to insert the contact: 2
```

```
Enter your choice: 4
Contact list (forward): Naveen  <-> Manu <-> Livins <-> Sujin <-> NULL
Contact list (backward): Sujin <-> Livins <-> Manu <-> Naveen  <-> NULL
```

```
Enter your choice: 3
Delete by name or position? (n/p): p
Enter the position (0-based index) to delete the contact: 1
```

```
Enter your choice: 5
Enter the contact's name to search: Livins
Livins found at position 1
```

```
Enter your choice: 6
Exiting the program...
```