**ASSIGNMENT-3**

**Name:Sivagovindan**

**Reg no:RA2311003050377**

**Year & Branch: II B.Tech CSE - F Section**

**Assignment No: 3**

**Github profile:** [**DSA-ASSIGNMENT-3/ at main · Sivagovindan4/DSA-ASSIGNMENT-3 (github.com)**](https://github.com/Sivagovindan4/DSA-ASSIGNMENT-3/tree/main)

**1. You are given a task of implementing a simple contact management system using a singly linked list. The system will manage contact names. Implement the following operations using a singly linked list and switch case. After every operation, display thecurrent list of contacts.**

**The operations to implement are:**

**(i) Creation of the list: Allow the user to create a list of contact names by entering them one by one.**

**(ii) Insertion of a new contact: Insert a new contact's name into a specific position**

**in the list. The user should provide the name and the position at which it should be inserted.**

**(iii) Deletion of a contact: Delete a contact's name from the list based on their position or name. Ask the user whether they want to delete by name or by position.**

**(iv) Traversal of the list: Display all the contact names in the list in the current order.**

**(v) Search for a contact: Search for a contact's name in the list and display whether or not the contact is found, along with their position if present.**

**Code:**

**#include <stdio.h>**

**#include <stdlib.h>**

**#include <string.h>**

**struct Node {**

**char name[100];**

**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 displayContacts(struct Node\* head) {**

**if (head == NULL) {**

**printf("The contact list is empty.\n");**

**return;**

**}**

**struct Node\* temp = head;**

**printf("Contacts in the list:\n");**

**while (temp != NULL) {**

**printf("%s\n", temp->name);**

**temp = temp->next;**

**}**

**}**

**void insertContact(struct Node\*\* head, char\* name, int position) {**

**struct Node\* newNode = createNode(name);**

**if (position == 1) {**

**newNode->next = \*head;**

**\*head = newNode;**

**} else {**

**struct Node\* temp = \*head;**

**for (int i = 1; temp != NULL && i < position - 1; i++) {**

**temp = temp->next;**

**}**

**if (temp == NULL) {**

**printf("Position out of range. Adding at the end of the list.\n");**

**newNode->next = NULL;**

**struct Node\* last = \*head;**

**if (last == NULL) {**

**\*head = newNode;**

**} else {**

**while (last->next != NULL) {**

**last = last->next;**

**}**

**last->next = newNode;**

**}**

**} else {**

**newNode->next = temp->next;**

**temp->next = newNode;**

**}**

**}**

**displayContacts(\*head);**

**}**

**void deleteContactByPosition(struct Node\*\* head, int position) {**

**if (\*head == NULL) {**

**printf("The contact list is empty.\n");**

**return;**

**}**

**struct Node\* temp = \*head;**

**if (position == 1) {**

**\*head = temp->next;**

**free(temp);**

**} else {**

**struct Node\* prev = NULL;**

**for (int i = 1; temp != NULL && i < position; i++) {**

**prev = temp;**

**temp = temp->next;**

**}**

**if (temp == NULL) {**

**printf("Position out of range.\n");**

**} else {**

**prev->next = temp->next;**

**free(temp);**

**}**

**}**

**displayContacts(\*head);**

**}**

**void deleteContactByName(struct Node\*\* head, char\* name) {**

**if (\*head == NULL) {**

**printf("The contact list is empty.\n");**

**return;**

**}**

**struct Node\* temp = \*head;**

**struct Node\* prev = NULL;**

**if (strcmp(temp->name, name) == 0) {**

**\*head = temp->next;**

**free(temp);**

**displayContacts(\*head);**

**return;**

**}**

**while (temp != NULL && strcmp(temp->name, name) != 0) {**

**prev = temp;**

**temp = temp->next;**

**}**

**if (temp == NULL) {**

**printf("Contact not found.\n");**

**} else {**

**prev->next = temp->next;**

**free(temp);**

**}**

**displayContacts(\*head);**

**}**

**void searchContact(struct Node\* head, char\* name) {**

**struct Node\* temp = head;**

**int position = 1;**

**while (temp != NULL) {**

**if (strcmp(temp->name, name) == 0) {**

**printf("Contact '%s' found at position %d.\n", name, position);**

**return;**

**}**

**temp = temp->next;**

**position++;**

**}**

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

**}**

**int main() {**

**struct Node\* head = NULL;**

**int choice, position;**

**char name[100];**

**while (1) {**

**printf("\n--- Contact Management System ---\n");**

**printf("1. Create Contact List\n");**

**printf("2. Insert New Contact\n");**

**printf("3. Delete Contact\n");**

**printf("4. Display Contacts\n");**

**printf("5. Search for a Contact\n");**

**printf("6. Exit\n");**

**printf("Enter your choice: ");**

**scanf("%d", &choice);**

**switch (choice) {**

**case 1:**

**printf("Enter contact name (or 'done' to finish): ");**

**while (1) {**

**scanf("%s", name);**

**if (strcmp(name, "done") == 0)**

**break;**

**insertContact(&head, name, 1);**

**}**

**break;**

**case 2:**

**printf("Enter contact name to insert: ");**

**scanf("%s", name);**

**printf("Enter the position to insert the contact: ");**

**scanf("%d", &position);**

**insertContact(&head, name, position);**

**break;**

**case 3:**

**printf("Delete by 1) Position or 2) Name: ");**

**int deleteChoice;**

**scanf("%d", &deleteChoice);**

**if (deleteChoice == 1) {**

**printf("Enter position to delete: ");**

**scanf("%d", &position);**

**deleteContactByPosition(&head, position);**

**} else if (deleteChoice == 2) {**

**printf("Enter name to delete: ");**

**scanf("%s", name);**

**deleteContactByName(&head, name);**

**} else {**

**printf("Invalid choice.\n");**

**}**

**break;**

**case 4:**

**displayContacts(head);**

**break;**

**case 5:**

**printf("Enter name to search: ");**

**scanf("%s", name);**

**searchContact(head, name);**

**break;**

**case 6:**

**printf("Exiting...\n");**

**return 0;**

**default:**

**printf("Invalid choice.\n");**

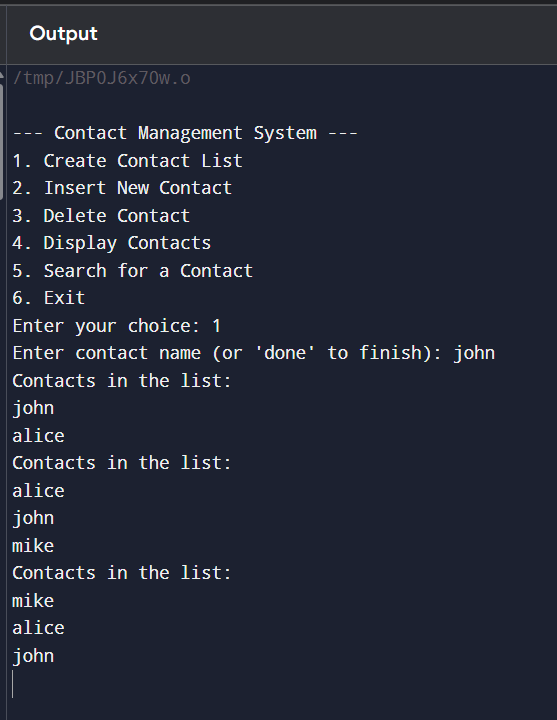
**}**

**}**

**return 0;**

**}**

**Output:**

****

**2. You are tasked with implementing a simple contact management system using a doubly linked list. The system will manage contact names. Implement the following operations using a doubly linked list and switch-case. After every operation, display the current list of contacts.**

**The operations to implement are:**

**(i) Creation of the list: Allow the user to create a list of contact names by entering them one by one.**

**(ii) Insertion of a new contact: Insert a new contact’s name into a specific position in the list. The user should provide the name and the position at which it should be inserted.**

**(iii)Deletion of a contact: Delete a contact’s name from the list based on their position or name. Ask the user whether they want to delete by name or by position.**

**(iv)Traversal of the list (in both directions): Display all the contact names in the list in the current order (forward traversal) and then display them in reverse order (backward traversal).**

**(v) Search for a contact: Search for a contact’s name in the list and display whether or not the contact is found, along with their position if present.**

**Code:**

**#include <stdio.h>**

**#include <stdlib.h>**

**#include <string.h>**

**struct Node {**

**char name[100];**

**struct Node\* next;**

**struct Node\* prev;**

**};**

**struct Node\* createNode(char\* name) {**

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

**strcpy(newNode->name, name);**

**newNode->next = NULL;**

**newNode->prev = NULL;**

**return newNode;**

**}**

**void displayContactsForward(struct Node\* head) {**

**if (head == NULL) {**

**printf("The contact list is empty.\n");**

**return;**

**}**

**struct Node\* temp = head;**

**printf("Contact list (forward): ");**

**while (temp != NULL) {**

**printf("%s <-> ", temp->name);**

**temp = temp->next;**

**}**

**printf("NULL\n");**

**}**

**void displayContactsBackward(struct Node\* tail) {**

**if (tail == NULL) {**

**printf("The contact list is empty.\n");**

**return;**

**}**

**struct Node\* temp = tail;**

**printf("Contact list (backward): ");**

**while (temp != NULL) {**

**printf("%s <-> ", temp->name);**

**temp = temp->prev;**

**}**

**printf("NULL\n");**

**}**

**void insertContact(struct Node\*\* head, struct Node\*\* tail, char\* name, int position) {**

**struct Node\* newNode = createNode(name);**

**if (position == 0) {**

**newNode->next = \*head;**

**if (\*head != NULL) {**

**(\*head)->prev = newNode;**

**}**

**\*head = newNode;**

**if (\*tail == NULL) {**

**\*tail = newNode;**

**}**

**} else {**

**struct Node\* temp = \*head;**

**for (int i = 0; temp != NULL && i < position - 1; i++) {**

**temp = temp->next;**

**}**

**if (temp == NULL || temp->next == NULL) {**

**newNode->prev = \*tail;**

**if (\*tail != NULL) {**

**(\*tail)->next = newNode;**

**}**

**\*tail = newNode;**

**} else {**

**newNode->next = temp->next;**

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

**temp->next = newNode;**

**newNode->prev = temp;**

**}**

**}**

**displayContactsForward(\*head);**

**displayContactsBackward(\*tail);**

**}**

**void deleteContactByPosition(struct Node\*\* head, struct Node\*\* tail, int position) {**

**if (\*head == NULL) {**

**printf("The contact list is empty.\n");**

**return;**

**}**

**struct Node\* temp = \*head;**

**if (position == 0) {**

**\*head = temp->next;**

**if (\*head != NULL) {**

**(\*head)->prev = NULL;**

**} else {**

**\*tail = NULL;**

**}**

**free(temp);**

**} else {**

**for (int i = 0; temp != NULL && i < position; i++) {**

**temp = temp->next;**

**}**

**if (temp == NULL) {**

**printf("Position out of range.\n");**

**return;**

**}**

**if (temp->next != NULL) {**

**temp->next->prev = temp->prev;**

**} else {**

**\*tail = temp->prev;**

**}**

**if (temp->prev != NULL) {**

**temp->prev->next = temp->next;**

**}**

**free(temp);**

**}**

**displayContactsForward(\*head);**

**displayContactsBackward(\*tail);**

**}**

**void deleteContactByName(struct Node\*\* head, struct Node\*\* tail, char\* name) {**

**if (\*head == NULL) {**

**printf("The contact list is empty.\n");**

**return;**

**}**

**struct Node\* temp = \*head;**

**while (temp != NULL && strcmp(temp->name, name) != 0) {**

**temp = temp->next;**

**}**

**if (temp == NULL) {**

**printf("Contact not found.\n");**

**return;**

**}**

**if (temp == \*head) {**

**\*head = temp->next;**

**}**

**if (temp == \*tail) {**

**\*tail = temp->prev;**

**}**

**if (temp->next != NULL) {**

**temp->next->prev = temp->prev;**

**}**

**if (temp->prev != NULL) {**

**temp->prev->next = temp->next;**

**}**

**free(temp);**

**displayContactsForward(\*head);**

**displayContactsBackward(\*tail);**

**}**

**void searchContact(struct Node\* head, char\* name) {**

**struct Node\* temp = head;**

**int position = 0;**

**while (temp != NULL) {**

**if (strcmp(temp->name, name) == 0) {**

**printf("Contact '%s' found at position %d.\n", name, position);**

**return;**

**}**

**temp = temp->next;**

**position++;**

**}**

**printf("Contact '%s' not found.\n", name);**

**}**

**int main() {**

**struct Node\* head = NULL;**

**struct Node\* tail = NULL;**

**int choice, position;**

**char name[100];**

**while (1) {**

**printf("\n--- Contact Management System ---\n");**

**printf("1. Create Contact List\n");**

**printf("2. Insert New Contact\n");**

**printf("3. Delete 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:**

**printf("Enter the number of contacts: ");**

**int numContacts;**

**scanf("%d", &numContacts);**

**for (int i = 0; i < numContacts; i++) {**

**printf("Enter contact name %d: ", i + 1);**

**scanf("%s", name);**

**insertContact(&head, &tail, name, i);**

**}**

**break;**

**case 2:**

**printf("Enter the contact's name to insert: ");**

**scanf("%s", name);**

**printf("Enter the position (0-based index) to insert the contact: ");**

**scanf("%d", &position);**

**insertContact(&head, &tail, name, position);**

**break;**

**case 3:**

**printf("Delete by name or position? (n/p): ");**

**char deleteChoice;**

**scanf(" %c", &deleteChoice);**

**if (deleteChoice == 'p') {**

**printf("Enter position to delete (0-based index): ");**

**scanf("%d", &position);**

**deleteContactByPosition(&head, &tail, position);**

**} else if (deleteChoice == 'n') {**

**printf("Enter name to delete: ");**

**scanf("%s", name);**

**deleteContactByName(&head, &tail, name);**

**} else {**

**printf("Invalid choice.\n");**

**}**

**break;**

**case 4:**

**displayContactsForward(head);**

**displayContactsBackward(tail);**

**break;**

**case 5:**

**printf("Enter name to search: ");**

**scanf("%s", name);**

**searchContact(head, name);**

**break;**

**case 6:**

**printf("Exiting the program...\n");**

**return 0;**

**default:**

**printf("Invalid choice.\n");**

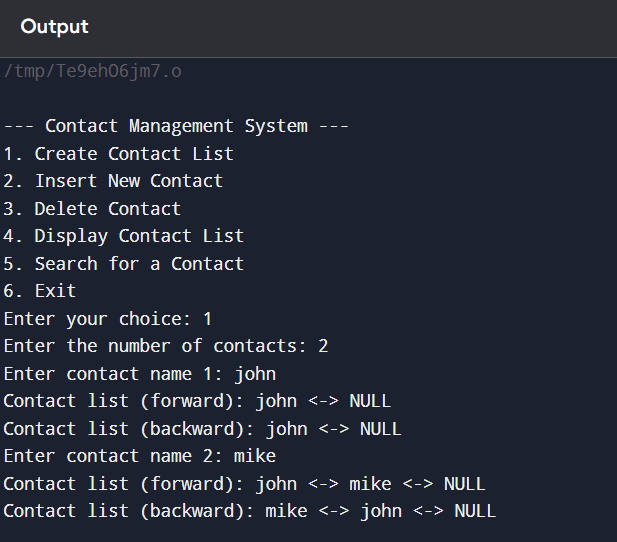
**}**

**}**

**return 0;**

**}**

**Output:**

****