**Assignment-4**

* Question no. 1:
* **Source Code:**

#include <stdio.h>

#include <stdlib.h>

// Node structure for a doubly linked list

struct Node {

int data;

struct Node\* prev;

struct Node\* next;

};

// Function prototypes

struct Node\* createNode(int data);

void displayList(struct Node\* head);

struct Node\* insertAtBeginning(struct Node\* head, int data);

struct Node\* insertAtEnd(struct Node\* head, int data);

struct Node\* insertBeforeNode(struct Node\* head, int data, int key);

struct Node\* insertAfterNode(struct Node\* head, int data, int key);

struct Node\* deleteFromBeginning(struct Node\* head);

struct Node\* deleteFromEnd(struct Node\* head);

struct Node\* deleteAfterNode(struct Node\* head, int key);

struct Node\* deleteList(struct Node\* head);

int main() {

struct Node\* head = NULL;

int choice, data, key;

do {

printf("\nDoubly Linked List Menu:\n");

printf("1. Create a doubly linked list\n");

printf("2. Display the list\n");

printf("3. Insert at the beginning\n");

printf("4. Insert at the end\n");

printf("5. Insert before a given node\n");

printf("6. Insert after a given node\n");

printf("7. Delete from the beginning\n");

printf("8. Delete from the end\n");

printf("9. Delete after a given node\n");

printf("10. Delete the entire list\n");

printf("0. Exit\n");

printf("Enter your choice: ");

scanf("%d", &choice);

switch (choice) {

case 1:

// Create a doubly linked list

printf("Enter data for the first node: ");

scanf("%d", &data);

head = createNode(data);

break;

case 2:

// Display the elements of the doubly linked list

displayList(head);

break;

case 3:

// Insert at the beginning

printf("Enter data for the new node: ");

scanf("%d", &data);

head = insertAtBeginning(head, data);

break;

case 4:

// Insert at the end

printf("Enter data for the new node: ");

scanf("%d", &data);

head = insertAtEnd(head, data);

break;

case 5:

// Insert before a given node

printf("Enter data for the new node: ");

scanf("%d", &data);

printf("Enter the key value before which to insert: ");

scanf("%d", &key);

head = insertBeforeNode(head, data, key);

break;

case 6:

// Insert after a given node

printf("Enter data for the new node: ");

scanf("%d", &data);

printf("Enter the key value after which to insert: ");

scanf("%d", &key);

head = insertAfterNode(head, data, key);

break;

case 7:

// Delete from the beginning

head = deleteFromBeginning(head);

break;

case 8:

// Delete from the end

head = deleteFromEnd(head);

break;

case 9:

// Delete after a given node

printf("Enter the key value after which to delete: ");

scanf("%d", &key);

head = deleteAfterNode(head, key);

break;

case 10:

// Delete the entire list

head = deleteList(head);

break;

case 0:

// Exit the program

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

break;

default:

printf("Invalid choice. Please enter a valid option.\n");

}

} while (choice != 0);

return 0;

}

// Function to create a new node

struct Node\* createNode(int data) {

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

newNode->data = data;

newNode->prev = NULL;

newNode->next = NULL;

return newNode;

}

// Function to display the doubly linked list

void displayList(struct Node\* head) {

printf("Doubly Linked List: ");

while (head != NULL) {

printf("%d <-> ", head->data);

head = head->next;

}

printf("NULL\n");

}

// Function to insert a node at the beginning

struct Node\* insertAtBeginning(struct Node\* head, int data) {

struct Node\* newNode = createNode(data);

if (head == NULL) {

return newNode;

}

newNode->next = head;

head->prev = newNode;

return newNode;

}

// Function to insert a node at the end

struct Node\* insertAtEnd(struct Node\* head, int data) {

struct Node\* newNode = createNode(data);

if (head == NULL) {

return newNode;

}

struct Node\* temp = head;

while (temp->next != NULL) {

temp = temp->next;

}

temp->next = newNode;

newNode->prev = temp;

return head;

}

// Function to insert a node before a given node

struct Node\* insertBeforeNode(struct Node\* head, int data, int key) {

struct Node\* newNode = createNode(data);

if (head == NULL) {

return newNode;

}

if (head->data == key) {

newNode->next = head;

head->prev = newNode;

return newNode;

}

struct Node\* temp = head;

while (temp->next != NULL && temp->next->data != key) {

temp = temp->next;

}

if (temp->next == NULL) {

printf("Key not found in the list.\n");

free(newNode);

return head;

}

newNode->next = temp->next;

newNode->prev = temp;

temp->next->prev = newNode;

temp->next = newNode;

return head;

}

// Function to insert a node after a given node

struct Node\* insertAfterNode(struct Node\* head, int data, int key) {

struct Node\* newNode = createNode(data);

if (head == NULL) {

return newNode;

}

struct Node\* temp = head;

while (temp != NULL && temp->data != key) {

temp = temp->next;

}

if (temp == NULL) {

printf("Key not found in the list.\n");

free(newNode);

return head;

}

newNode->next = temp->next;

if (temp->next != NULL) {

temp->next->prev = newNode;

}

newNode->prev = temp;

temp->next = newNode;

return head;

}

// Function to delete a node from the beginning

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

if (head == NULL) {

printf("List is empty. Nothing to delete.\n");

return NULL;

}

struct Node\* newHead = head->next;

if (newHead != NULL) {

newHead->prev = NULL;

}

free(head);

return newHead;

}

// Function to delete a node from the end

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

if (head == NULL) {

printf("List is empty. Nothing to delete.\n");

return NULL;

}

if (head->next == NULL) {

free(head);

return NULL;

}

struct Node\* temp = head;

while (temp->next->next != NULL) {

temp = temp->next;

}

free(temp->next);

temp->next = NULL;

return head;

}

// Function to delete a node after a given node

struct Node\* deleteAfterNode(struct Node\* head, int key) {

if (head == NULL) {

printf("List is empty. Nothing to delete.\n");

return NULL;

}

struct Node\* temp = head;

while (temp != NULL && temp->data != key) {

temp = temp->next;

}

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

printf("Key not found or no node after the key to delete.\n");

return head;

}

struct Node\* nodeToDelete = temp->next;

temp->next = nodeToDelete->next;

if (nodeToDelete->next != NULL) {

nodeToDelete->next->prev = temp;

}

free(nodeToDelete);

return head;

}

// Function to delete the entire list

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

while (head != NULL) {

struct Node\* temp = head;

head = head->next;

free(temp);

}

printf("Entire list deleted.\n");

return NULL;

}

* **Output:**

Doubly Linked List Menu:

1. Create a doubly linked list

2. Display the list

3. Insert at the beginning

4. Insert at the end

5. Insert before a given node

6. Insert after a given node

7. Delete from the beginning

8. Delete from the end

9. Delete after a given node

10. Delete the entire list

0. Exit

Enter your choice: 1

Enter data for the first node: 10

Doubly Linked List Menu:

1. Create a doubly linked list

2. Display the list

3. Insert at the beginning

4. Insert at the end

5. Insert before a given node

6. Insert after a given node

7. Delete from the beginning

8. Delete from the end

9. Delete after a given node

10. Delete the entire list

0. Exit

Enter your choice: 2

Doubly Linked List: 10 <-> NULL

Doubly Linked List Menu:

1. Create a doubly linked list

2. Display the list

3. Insert at the beginning

4. Insert at the end

5. Insert before a given node

6. Insert after a given node

7. Delete from the beginning

8. Delete from the end

9. Delete after a given node

10. Delete the entire list

0. Exit

Enter your choice: 3

Enter data for the new node: 45

Doubly Linked List Menu:

1. Create a doubly linked list

2. Display the list

3. Insert at the beginning

4. Insert at the end

5. Insert before a given node

6. Insert after a given node

7. Delete from the beginning

8. Delete from the end

9. Delete after a given node

10. Delete the entire list

0. Exit

Enter your choice: 2

Doubly Linked List: 45 <-> 10 <-> NULL

Doubly Linked List Menu:

1. Create a doubly linked list

2. Display the list

3. Insert at the beginning

4. Insert at the end

5. Insert before a given node

6. Insert after a given node

7. Delete from the beginning

8. Delete from the end

9. Delete after a given node

10. Delete the entire list

0. Exit

Enter your choice: 4

Enter data for the new node: 50

Doubly Linked List Menu:

1. Create a doubly linked list

2. Display the list

3. Insert at the beginning

4. Insert at the end

5. Insert before a given node

6. Insert after a given node

7. Delete from the beginning

8. Delete from the end

9. Delete after a given node

10. Delete the entire list

0. Exit

Enter your choice: 2

Doubly Linked List: 45 <-> 10 <-> 50 <-> NULL

Doubly Linked List Menu:

1. Create a doubly linked list

2. Display the list

3. Insert at the beginning

4. Insert at the end

5. Insert before a given node

6. Insert after a given node

7. Delete from the beginning

8. Delete from the end

9. Delete after a given node

10. Delete the entire list

0. Exit

Enter your choice: 5

Enter data for the new node: 34

Enter the key value before which to insert: 10

Doubly Linked List Menu:

1. Create a doubly linked list

2. Display the list

3. Insert at the beginning

4. Insert at the end

5. Insert before a given node

6. Insert after a given node

7. Delete from the beginning

8. Delete from the end

9. Delete after a given node

10. Delete the entire list

0. Exit

Enter your choice: 2

Doubly Linked List: 45 <-> 34 <-> 10 <-> 50 <-> NULL

Doubly Linked List Menu:

1. Create a doubly linked list

2. Display the list

3. Insert at the beginning

4. Insert at the end

5. Insert before a given node

6. Insert after a given node

7. Delete from the beginning

8. Delete from the end

9. Delete after a given node

10. Delete the entire list

0. Exit

Enter your choice: 6

Enter data for the new node: 22

Enter the key value after which to insert: 50

Doubly Linked List Menu:

1. Create a doubly linked list

2. Display the list

3. Insert at the beginning

4. Insert at the end

5. Insert before a given node

6. Insert after a given node

7. Delete from the beginning

8. Delete from the end

9. Delete after a given node

10. Delete the entire list

0. Exit

Enter your choice: 2

Doubly Linked List: 45 <-> 34 <-> 10 <-> 50 <-> 22 <-> NULL

Doubly Linked List Menu:

1. Create a doubly linked list

2. Display the list

3. Insert at the beginning

4. Insert at the end

5. Insert before a given node

6. Insert after a given node

7. Delete from the beginning

8. Delete from the end

9. Delete after a given node

10. Delete the entire list

0. Exit

Enter your choice: 7

Doubly Linked List Menu:

1. Create a doubly linked list

2. Display the list

3. Insert at the beginning

4. Insert at the end

5. Insert before a given node

6. Insert after a given node

7. Delete from the beginning

8. Delete from the end

9. Delete after a given node

10. Delete the entire list

0. Exit

Enter your choice: 2

Doubly Linked List: 34 <-> 10 <-> 50 <-> 22 <-> NULL

Doubly Linked List Menu:

1. Create a doubly linked list

2. Display the list

3. Insert at the beginning

4. Insert at the end

5. Insert before a given node

6. Insert after a given node

7. Delete from the beginning

8. Delete from the end

9. Delete after a given node

10. Delete the entire list

0. Exit

Enter your choice: 8

Doubly Linked List Menu:

1. Create a doubly linked list

2. Display the list

3. Insert at the beginning

4. Insert at the end

5. Insert before a given node

6. Insert after a given node

7. Delete from the beginning

8. Delete from the end

9. Delete after a given node

10. Delete the entire list

0. Exit

Enter your choice: 2

Doubly Linked List: 34 <-> 10 <-> 50 <-> NULL

Doubly Linked List Menu:

1. Create a doubly linked list

2. Display the list

3. Insert at the beginning

4. Insert at the end

5. Insert before a given node

6. Insert after a given node

7. Delete from the beginning

8. Delete from the end

9. Delete after a given node

10. Delete the entire list

0. Exit

Enter your choice: 9

Enter the key value after which to delete: 10

Doubly Linked List Menu:

1. Create a doubly linked list

2. Display the list

3. Insert at the beginning

4. Insert at the end

5. Insert before a given node

6. Insert after a given node

7. Delete from the beginning

8. Delete from the end

9. Delete after a given node

10. Delete the entire list

0. Exit

Enter your choice: 2

Doubly Linked List: 34 <-> 10 <-> NULL

Doubly Linked List Menu:

1. Create a doubly linked list

2. Display the list

3. Insert at the beginning

4. Insert at the end

5. Insert before a given node

6. Insert after a given node

7. Delete from the beginning

8. Delete from the end

9. Delete after a given node

10. Delete the entire list

0. Exit

Enter your choice: 10

Entire list deleted.

Doubly Linked List Menu:

1. Create a doubly linked list

2. Display the list

3. Insert at the beginning

4. Insert at the end

5. Insert before a given node

6. Insert after a given node

7. Delete from the beginning

8. Delete from the end

9. Delete after a given node

10. Delete the entire list

0. Exit

Enter your choice: 2

Doubly Linked List: NULL

Doubly Linked List Menu:

1. Create a doubly linked list

2. Display the list

3. Insert at the beginning

4. Insert at the end

5. Insert before a given node

6. Insert after a given node

7. Delete from the beginning

8. Delete from the end

9. Delete after a given node

10. Delete the entire list

0. Exit

Enter your choice: 0

Exiting the program.