WAP to Implement Singly Linked List with following operations a) Create a linked list. b) Deletion of first element, specified element and last element in the list. c) Display the contents of the linked list.

```
#include <stdio.h>
#include <stdlib.h>
struct Node {
int data;
struct Node *next;
};
struct Node *head = NULL;
void create(int value) {
struct Node *newNode = (struct Node*)malloc(sizeof(struct Node));
newNode->data = value;
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.\n");
}
void deleteFirst() {
if(head == NULL) {
printf("List is empty.\n");
return;
}
struct Node *temp = head;
head = head->next;
printf("Deleted: %d\n", temp->data);
free(temp);
}
void deleteSpecific(int key) {
if(head == NULL) {
printf("List is empty.\n");
return;
}
struct Node *temp = head, *prev = NULL;
if(temp != NULL && temp->data == key) {
head = temp->next;
printf("Deleted: %d\n", key);
free(temp);
return;
}
while(temp != NULL && temp->data != key) {
prev = temp;
temp = temp->next;
}
if(temp == NULL) {
printf("%d not found in list.\n", key);
return;
```

```
}
prev->next = temp->next;
printf("Deleted: %d\n", key);
free(temp);
}
void deleteLast() {
if(head == NULL) {
printf("List is empty.\n");
return;
}
if(head->next == NULL) {
printf("Deleted: %d\n", head->data);
free(head);
head = NULL;
return;
}
struct Node *temp = head;
while(temp->next->next != NULL)
temp = temp->next;
printf("Deleted: %d\n", temp->next->data);
free(temp->next);
temp->next = NULL;
}
void display() {
struct Node *temp = head;
if(temp == NULL) {
printf("List is empty.\n");
return;
}
printf("Linked List: ");
while(temp != NULL) {
printf("%d -> ", temp->data);
temp = temp->next;
}
printf("NULL\n");
}
int main() {
int choice, val;
while(1) {
printf("\n--- MENU ---\n");
printf("1. Create Node\n2. Delete First\n3. Delete Specific\n4. Delete Last\n5. Display\n6. Exit\n");
printf("Enter your choice: ");
scanf("%d", &choice);
switch(choice) {
case 1: printf("Enter value: ");
scanf("%d", &val);
create(val); break;
case 2: deleteFirst(); break;
case 3: printf("Enter value to delete: ");
scanf("%d", &val);
deleteSpecific(val); break;
case 4: deleteLast(); break;
case 5: display(); break;
```

```
case 6: exit(0);
default: printf("Invalid choice!\n");
}
}
return 0;
```

```
--- MENU ---
1. Create Node
2. Delete First
3. Delete Specific
4. Delete Last
5. Display
6. Exit
Enter your choice: 1
Enter value: 10
Node inserted.

--- MENU ---
1. Create Node
2. Delete First
3. Delete Specific
4. Delete Last
5. Display
6. Exit
Enter your choice: 1
Enter value: 20
Node inserted.

--- MENU ---
1. Create Node
2. Delete First
3. Delete Specific
4. Delete Last
5. Display
6. Exit
Enter your choice: 1
Enter value: 30
Node inserted.

--- MENU ---
1. Create Node
2. Delete First
3. Delete Specific
4. Delete Last
5. Display
6. Exit
Enter your choice: 1
Enter value: 40
Node inserted.
```

```
--- MENU ---
1. Create Node
2. Delete First
3. Delete Specific
4. Delete Last
5. Display
6. Exit
Enter your choice: 2
Deleted: 10

--- MENU ---
1. Create Node
2. Delete First
3. Delete Specific
4. Delete Last
5. Display
6. Exit
Enter your choice: 3
Enter value to delete: 30
Deleted: 30

--- MENU ---
1. Create Node
2. Delete First
3. Delete Specific
4. Delete Last
5. Display
6. Exit
Enter your choice: 5
Linked List: 20 -> 40 -> NULL
```