## Aim:- Implementation of Singly Linked List

## Code:-

```
#include <stdio.h>
#include <stdlib.h>
struct node
{ int data;
struct node *next;
};
struct node *start = NULL;
struct node *create(struct node *);
struct node *display(struct node *);
struct node *insertbeginning(struct node *);
struct node *insertend(struct node *);
struct node *insertmiddle(struct node *);
struct node *deletebeginning(struct node *);
struct node *deleteend(struct node *);
struct node *deletemiddle(struct node *);
int main(int argc, char *argv[]) {
int option;
printf("D10A_Atharva Chavan_9\n");
printf("****Implementation of Singly Linked List****\n");
do
{
printf("\n Choose any one of the following operations");
```

```
printf("\n 1: Create a list");
printf("\n 2: Display the list");
printf("\n 3: Add a node at the beginning");
printf("\n 4: Add a node at the end");
printf("\n 5: Add a node in the middle");
printf("\n 6: Delete a node from the beginning");
printf("\n 7: Delete a node from the end");
printf("\n 8: Delete a node after a given node");
printf("\n 9: Exit");
printf("\n Enter your option: ");
scanf("%d", &option);
switch(option)
{
case 1: start = create(start);
printf("\n LINKED LIST CREATED");
printf("\n");
break;
case 2: start = display(start);
printf("\n");
break;
case 3: start = insertbeginning(start);
printf("\n");
break;
case 4: start = insertend(start);
printf("\n");
```

```
break;
case 5: start = insertmiddle(start);
printf("\n");
break;
case 6: start = deletebeginning(start);
printf("\n");
break;
case 7: start = deleteend(start);
printf("\n");
break;
case 8: start = deletemiddle(start);
printf("\n");
break;
}
}while(option !=9);
printf("You have exited the linked list!");
printf("\n");
return 0;
}
struct node *create(struct node *start)
{
struct node *new_node, *ptr;
int num;
printf("\n Enter -1 to end");
printf("\n Enter the data : ");
```

```
scanf("%d", &num);
while(num!=-1)
{
new_node = (struct node*)malloc(sizeof(struct node));
new_node -> data=num;
if(start==NULL)
{
new_node -> next = NULL;
start = new_node;
}
else
{
ptr=start;
while(ptr->next!=NULL)
ptr=ptr->next;
ptr->next = new_node;
new_node->next=NULL;
}
printf("\n Enter the data : ");
scanf("%d", &num);
} return start;
}
struct node *display(struct node *start)
{
struct node *ptr;
```

```
ptr = start;
while(ptr != NULL)
{
printf("\t %d", ptr -> data);
ptr = ptr -> next;
} return start;
}
struct node *insertbeginning(struct node *start)
{
struct node *new_node;
int num;
printf("\n Enter the data : ");
scanf("%d", &num);
new_node = (struct node *)malloc(sizeof(struct node));
new_node -> data = num;
new_node -> next = start;
start = new_node;
return start;
}
struct node *insertend(struct node *start)
{
struct node *ptr, *new_node;
int num;
printf("\n Enter the data : ");
scanf("%d", &num);
```

```
new_node = (struct node *)malloc(sizeof(struct node));
new_node -> data = num;
new_node -> next = NULL;
ptr = start;
while(ptr -> next != NULL)
ptr = ptr -> next;
ptr -> next = new_node;
return start;
}
struct node *insertmiddle(struct node *start)
{
struct node *new_node, *ptr, *preptr;
int num, val;
printf("\n Enter the data : ");
scanf("%d", &num);
printf("\n Enter the value after which the data has to be inserted : ");
scanf("%d", &val);
new_node = (struct node *)malloc(sizeof(struct node));
new_node -> data = num;
ptr = start;
preptr = ptr;
while(preptr -> data != val)
{
preptr = ptr;
ptr = ptr -> next;
```

```
}
preptr -> next=new_node;
new_node -> next = ptr;
return start;
}
struct node *deletebeginning(struct node *start)
{
struct node *ptr;
ptr = start;
start = start -> next;
free(ptr);
return start;
}
struct node *deleteend(struct node *start)
{
struct node *ptr, *preptr;
ptr = start;
while(ptr -> next != NULL)
{
preptr = ptr;
ptr = ptr -> next;
}
preptr -> next = NULL;
free(ptr);
return start;
```

```
}
struct node *deletemiddle(struct node *start)
{
struct node *ptr, *preptr;
int val;
printf("\n Enter the value of the node which has to be deleted : ");
scanf("%d", &val);
ptr = start;
if(ptr -> data == val)
{
start = deletebeginning(start);
return start;
}
else
{
while(ptr -> data != val)
{
preptr = ptr;
ptr = ptr -> next;
}
preptr -> next = ptr -> next;
free(ptr);
return start;
}}
```

Output:-

```
/tmp/KZAJFyJRYm.o
D10A_Atharva Chavan_9
****Implementation of Singly Linked List****

 Choose any one of the following operations
 1: Create a list
 2: Display the list
 3: Add a node at the beginning
 4: Add a node at the end
 5: Add a node in the middle
6: Delete a node from the beginning
 7: Delete a node from the end
 8: Delete a node after a given node
 9: Exit
 Enter your option: 1
 Enter -1 to end
 Enter the data : 10
 Enter the data : 20
 Enter the data : 30
 Enter the data : 40
 Enter the data : -1
 LINKED LIST CREATED

 Choose any one of the following operations
 1: Create a list
 2: Display the list
 3: Add a node at the beginning
 4: Add a node at the end
 5: Add a node in the middle
 6: Delete a node from the beginning
 7: Delete a node from the end
```

```
8: Delete a node after a given node
9: Exit
Enter your option: 3
Enter the data : 0
Choose any one of the following operations
1: Create a list
2: Display the list
3: Add a node at the beginning
4: Add a node at the end
5: Add a node in the middle
6: Delete a node from the beginning
7: Delete a node from the end
8: Delete a node after a given node
9: Exit
Enter your option: 4
Enter the data : 50
Choose any one of the following operations
1: Create a list
2: Display the list
3: Add a node at the beginning
4: Add a node at the end
5: Add a node in the middle
6: Delete a node from the beginning
7: Delete a node from the end
8: Delete a node after a given node
9: Exit
Enter your option: 6
Choose any one of the following operations
1: Create a list
2: Display the list
3: Add a node at the beginning
```

```
4: Add a node at the end
5: Add a node in the middle
6: Delete a node from the beginning
7: Delete a node from the end
8: Delete a node after a given node
9: Exit
Enter your option: 7
Choose any one of the following operations
1: Create a list
2: Display the list
3: Add a node at the beginning
4: Add a node at the end
5: Add a node in the middle
6: Delete a node from the beginning
7: Delete a node from the end
8: Delete a node after a given node
9: Exit
Enter your option: 2
10  20  30  40

Choose any one of the following operations
1: Create a list
2: Display the list
3: Add a node at the beginning
4: Add a node at the end
5: Add a node in the middle
6: Delete a node from the beginning
7: Delete a node from the end
8: Delete a node after a given node
9: Exit
Enter your option: 9
You have exited the linked list!
```