Linked List

1.Simple Linked List:

10 → 20 → 30→ 40 → NULL

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

struct node{
    int data;
    struct node *next;
};

void print(struct node *p){
    printf("\nYour Linked List are : ");
    while(p != NULL){
        printf("%d\t", p->data);
        p = p->next;
    }
}

int main(){
    //Memory allocation
    struct node *head  = malloc(sizeof(struct node));
    struct node *node2 = malloc(sizeof(struct node));
    struct node *node3 = malloc(sizeof(struct node));
    struct node *node4 = malloc(sizeof(struct node));

    //Data assign for each nodes
    head->data = 10;
    node2->data = 20;
    node3->data = 30;
    node4->data = 40;

   //Next address assign for each node
    head->next = node2;
    node2->next = node3;
    node3->next = node4;
    node4->next = NULL;

//then call print function to print all data
    print(head);
return 0;
}
```

2.Create Linked List with n number of Node?

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

struct node{
    int data;
    struct node *next;
};

void print(struct node *p){
    printf("\nYour Linked List are : ");
    while(p != NULL){
        printf("%d\t", p->data);
        p = p->next;
    }
}

void create_List(struct node *p){
    int n,x;
    printf("Enter Number of Node : ");
    scanf("%d", &n);
    for(int i=1; i<=n; i++){
        printf("Enter Node-%d data: ", i);
        scanf("%d", &x);
        p->data = x;

        if(i<n){
            struct node *temp = malloc(sizeof(struct node));
            p->next = temp;
            p = p->next;
        }
    }
    p->next = NULL;
}

int main(){
    //Memory allocation for head node then send it as parameter
    struct node *head  = malloc(sizeof(struct node));
    create_List(head);

    //then call print functin to print all data
    print(head);
return 0;
}
```

3.Create Linked List which contain multiple datatypes:

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

struct student{
    int roll;
    char name[20];
    float cgpa;
    struct student *next;
};

void print(struct student *p){
    printf("\nYour Linked List are : \n");
    while(p != NULL){
        printf("%d %s %.2f\n", p->roll, p->name, p->cgpa);
        p = p->next;
    }
}

void create_List(struct student *p){
    int n,x;
    char nm[20];
    float cg;
    printf("Enter Number of student : ");
    scanf("%d", &n);
    for(int i=1; i<=n; i++){
        printf("Enter student-%d data: ", i);
        scanf("%d %s %f", &x, nm, &cg);
        p->roll = x;
        strcpy(p->name, nm);
        p->cgpa = cg;

        if(i<n){
            struct student *temp = malloc(sizeof(struct student));
            p->next = temp;
            p = p->next;
        }
    }
    p->next = NULL;
}

int main(){
    struct student *head  = malloc(sizeof(struct student));
    create_List(head);
    print(head);
return 0;
}
```

4.Insert a Node at beginning:

20 → 30 → 40 → 50 → NULL;
10 → 20 → 30 → 40 → 50 → NULL;

```c
#include <stdio.h>
#include <stdlib.h>
struct node{
    int data;
    struct node *next;
};

void print(struct node *p){
    printf("\nYour Linked List are : ");
    while(p != NULL){
        printf("%d\t", p->data);
        p = p->next;
    }
}

void add_Node_at_Begining(struct node **p, int val){
    struct node *temp = malloc(sizeof(struct node));
    temp->data = val;
    temp->next = *p;
    *p = temp;
}

int main(){
    struct node *head  = malloc(sizeof(struct node));
    struct node *node2 = malloc(sizeof(struct node));
    struct node *node3 = malloc(sizeof(struct node));
    struct node *node4 = malloc(sizeof(struct node));

    head->data = 20;
    node2->data = 30;
    node3->data = 40;
    node4->data = 50;

    head->next = node2;
    node2->next = node3;
    node3->next = node4;
    node4->next = NULL;

    print(head);
    add_Node_at_Begining(&head, 10);
    print(head);
return 0;
}
```

5.Insert a Node at End:

10 → 20 → 30 → 40 → NULL;
10 → 20 → 30 → 40 → 50 → NULL;

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

struct node{
    int data;
    struct node *next;
};

void print(struct node *p){
    printf("\nYour Linked List are : ");
    while(p != NULL){
        printf("%d\t", p->data);
        p = p->next;
    }
}

void add_Node_at_End(struct node *p, int val){
    struct node *temp = malloc(sizeof(struct node));
    temp->data = val;
    while(p->next!=NULL) p = p->next;
    p->next = temp;
    temp->next = NULL;
}

int main(){
    struct node *head  = malloc(sizeof(struct node));
    struct node *node2 = malloc(sizeof(struct node));
    struct node *node3 = malloc(sizeof(struct node));
    struct node *node4 = malloc(sizeof(struct node));

    head->data = 10;
    node2->data = 20;
    node3->data = 30;
    node4->data = 40;

    head->next = node2;
    node2->next = node3;
    node3->next = node4;
    node4->next = NULL;

    print(head);
    add_Node_at_End(head, 50);
    print(head);
return 0;
}
```

6.Insert a Node at any place(without beginning):

10 → 20 → 30 → 40 → NULL

10 → 20 → 100 → 30 → 40 → NULL

```c
#include <stdio.h>
#include <stdlib.h>
struct node{
    int data;
    struct node *next;
};
void print(struct node *p){
    printf("\nYour Linked List are : ");
    while(p != NULL){
        printf("%d\t", p->data);
        p = p->next;
    }
}
void Insert(struct node *p, int n, int val){
    struct node *temp = malloc(sizeof(struct node));
    temp->data = val;
    int cnt=1;
    while(cnt!=n-1){
        p=p->next; ++cnt;
    }
    struct node *t = p->next;
    p->next = temp;
    temp->next = t;
}

int main(){
    struct node *head  = malloc(sizeof(struct node));
    struct node *node2 = malloc(sizeof(struct node));
    struct node *node3 = malloc(sizeof(struct node));
    struct node *node4 = malloc(sizeof(struct node));

    head->data = 10;
    node2->data = 20;
    node3->data = 30;
    node4->data = 40;

    head->next = node2;
    node2->next = node3;
    node3->next = node4;
    node4->next = NULL;

    print(head);
    Insert(head, 3, 100);
    print(head);
return 0;
}
```

7.Delete a Node:

10→20→30→40→NULL
10→20→40→NULL

```c
#include <stdio.h>
#include <stdlib.h>
struct node{
    int data;
    struct node *next;
};
void print(struct node *p){
    printf("\nYour Linked List are : ");
    while(p != NULL){
        printf("%d\t", p->data);
        p = p->next;
    }
}
void Delete(struct node **p, int val){
    struct node *t = *p;
    if(t->data==val){
        *p = t->next;
        free(t);
    }
    else{
        while(t->next->data!=val) t=t->next;
        struct node *del = t->next;
        t->next = t->next->next; free(del);
    }
}

int main(){
    struct node *head  = malloc(sizeof(struct node));
    struct node *node2 = malloc(sizeof(struct node));
    struct node *node3 = malloc(sizeof(struct node));
    struct node *node4 = malloc(sizeof(struct node));

    head->data = 10;
    node2->data = 20;
    node3->data = 30;
    node4->data = 40;

    head->next = node2;
    node2->next = node3;
    node3->next = node4;
    node4->next = NULL;

    print(head);
    Delete(&head, 30);
    print(head);
return 0;
}
```

8.Edit a Data:

10 → 15 → 30 → 40 → NULL
10 → 20 → 30 → 40 → NULL

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

struct node{
    int data;
    struct node *next;
};

void print(struct node *p){
    printf("\nYour Linked List are : ");
    while(p != NULL){
        printf("%d\t", p->data);
        p = p->next;
    }
}

void Edit(struct node *p, int old, int new){
    while(p!=NULL){
        if(p->data==old){
            p->data = new;
            break;
        }
        else p=p->next;
    }
}

int main(){
    struct node *head  = malloc(sizeof(struct node));
    struct node *node2 = malloc(sizeof(struct node));
    struct node *node3 = malloc(sizeof(struct node));
    struct node *node4 = malloc(sizeof(struct node));

    head->data = 10;
    node2->data = 15;
    node3->data = 30;
    node4->data = 40;

    head->next = node2;
    node2->next = node3;
    node3->next = node4;
    node4->next = NULL;

    print(head);
    Edit(head, 15, 20);
    print(head);
return 0;
}
```

9.Search a Data:

10 → 20 → 30 → 40 → NULL
20 : Found 50 : Not Found

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

struct node{
    int data;
    struct node *next;
};

void print(struct node *p){
    printf("\nYour Linked List are : ");
    while(p != NULL){
        printf("%d\t", p->data);
        p = p->next;
    }
}
void Search(struct node *p, int elem){
    int yes=0;
    while(p!=NULL){
        if(p->data==elem){
            printf("\n%d Found!", elem); yes=1; break;
        }
        else p=p->next;
    }
if(!yes) printf("\n%d Not Found!", elem);
}

int main(){
    struct node *head  = malloc(sizeof(struct node));
    struct node *node2 = malloc(sizeof(struct node));
    struct node *node3 = malloc(sizeof(struct node));
    struct node *node4 = malloc(sizeof(struct node));

    head->data = 10;
    node2->data = 20;
    node3->data = 30;
    node4->data = 40;

    head->next = node2;
    node2->next = node3;
    node3->next = node4;
    node4->next = NULL;

    print(head);
    Search(head, 20); // 20 Found!
    Search(head, 50); // 50 Not Found!
return 0;
}
```

Stack

Pop, Push, Peek, Display :

```c
#include <stdio.h>
#include <stdlib.h>
#define max_size 7
int Stack[max_size], top = -1;
int main();
void Push(){
    int val;
    printf("Enter Value: ");
    scanf("%d", &val);
    if(top==max_size-1) printf("Stack Overflow!");
    else{
        Stack[++top] = val;
        printf("Value Pushed!\n");
    }
}

void Pop(){
    if(top==-1) printf("Stack Underflow!\n");
    else{
        --top;
        printf("Poped!\n");
    }
}

void Peek(){
    if(top==-1) printf("Stack Underflow!\n");
    else{
        printf("%d Peeked!\n", Stack[top]);
    }
}
void Display(){
    if(top==-1) printf("Stack is Empty Now!\n");
    else{
        for(int i=top; i>=0; i--) printf("%d\n", Stack[i]);
    }
}
int main(){
    int n;
    printf("\n1.Push\t2.Pop\t3.Peek\t4.Display\t5.Exit\nChoose Your Choice: ");
    scanf("%d", &n);
    n==1? Push() : n==2? Pop() : n==3? Peek() : n==4? Display() : n==5? exit(0) :
printf("\nEnter Correct One!\t");
    main();
return 0;
}
```

*Stack Using Linked List:

```c
#include <stdio.h>
#include <stdlib.h>
#define max_size 5

struct node{
    int data;
    struct node *next;
};
void Push(struct node *p){
    int val;
    printf("Enter Value: ");
    scanf("%d", &val);
    if(p->data == -100){
        p->data = val;
        p->next = NULL;
        printf("----Value Pushed!----\n\n");
    }
    else{
        int cnt=1;
        while(p->next != NULL){
            ++cnt;
            p = p->next;
        }
        if(cnt==max_size) printf("Stack Overflow!\n\n");
        else{
            struct node *temp = malloc(sizeof(struct node));
            temp->data = val;
            temp->next = NULL;
            p->next = temp;
            printf("----Value Pushed!----\n\n");
        }
    }
}
void Pop(struct node *p){
    if(p->data==-100) printf("Stack Underflow!\n\n");
    else{
        if(p->next==NULL){
            printf("%d Poped\n\n", p->data);
            p->data = -100;
        }
        else{
            while(p->next->next!=NULL) p=p->next;
            printf("%d Poped\n\n", p->next->data);
            free(p->next->next);
            p->next = NULL;
        }
    }
}
```

```c
void Peek(struct node *p){
    if(p->data==-100) printf("Stack Underflow!\n\n");
    else{
        if(p->next==NULL){
            printf("%d Peeked!\n\n", p->data);
        }
        else{
            while(p->next!=NULL) p=p->next;
            printf("%d Peeked!\n\n", p->data);
        }
    }
}


void Display(struct node *p){
    if(p->data==-100) printf("Stack is Empty!\n\n");
    else{
        while(p!=NULL){
            printf("%d\t", p->data);
            p = p->next;
        }
        printf("\n");
    }
}

int main(){
    struct node *head = malloc(sizeof(struct node));
    head->data = -100;
    while(1){
        int n;
        printf("1.Push\t2.Pop\t3.Peek\t4.Display\t5.Exit\nEnter
Your Choice: ");
        scanf("%d", &n);
        if(n==1) Push(head);
        else if(n==2) Pop(head);
        else if(n==3) Peek(head);
        else if(n==4) Display(head);
        else if(n==5) exit(0);
        else printf("----Please Enter Correct One!-----\n\n");
    }
return 0;
}
```

Queue

Queue Pop,Push,Peek,Display:

```c
#include <stdio.h>
#include <stdlib.h>
#define max_size 5
int Queue[max_size], f=-1, r=-1;
int main();
void Push(){
    int val;
    printf("Enter Value: ");
    scanf("%d", &val);
    if((r-f+1 == max_size) || (r-f+1==0)) printf("Queue Overflow!");
    else{
        if(r==-1) f=0;
        if(r==max_size-1) r=-1;
        Queue[++r] = val;
        printf("Value Pushed!\n");
    }
}

void Pop(){
    if(f==-1) printf("Queue Underflow!\n");
    else{
        printf("%d Poped!\n", Queue[f]);
        for(int i=f; i<r; i++) Queue[i] =  Queue[i+1];
        --r;
    }
}
void Peek(){
    if(f==-1) printf("Queue Underflow!\n");
    else{
        printf("%d Peeked!\n", Queue[f]);
    }
}
void Display(){
    if(r==-1) printf("Queue is Empty Now!\n");
    else{
        for(int i=r; i>=f; i--) printf("%d\n", Queue[i]);
    }
}
int main(){
    int n;
    printf("\n1.Push\t2.Pop\t3.Peek\t4.Display\t5.Exit\nChoose Your Choice: ");
    scanf("%d", &n);
    n==1? Push() : n==2? Pop() : n==3? Peek() : n==4? Display() : n==5? exit(0) :
printf("\nEnter Correct One!\n");
    main();
return 0;
}
```

*Queue Using Linked List:

```c
#include <stdio.h>
#include <stdlib.h>
#define max_size 5

struct node{
    int data;
    struct node *next;
};

void Push(struct node *p){
    int val;
    printf("Enter Value: ");
    scanf("%d", &val);
    if(p->data == -100){
        p->data = val;
        p->next = NULL;
        printf("----Value Pushed!----\n\n");
    }
    else{
        int cnt=1;
        while(p->next != NULL){
            ++cnt;
            p = p->next;
        }
        if(cnt==max_size) printf("Queue Overflow!\n\n");
        else{
            struct node *temp = malloc(sizeof(struct node));
            temp->data = val;
            temp->next = NULL;
            p->next = temp;
            printf("----Value Pushed!----\n\n");
        }
    }
}
void Pop(struct node **p){
    struct node *temp = *p;
    if(temp->data==-100) printf("Queue Underflow!\n\n");
    else{
        printf("----%d Poped!----\n\n", temp->data);
        if(temp->next==NULL){
            temp->data = -100;
        }
        else{
            *p=temp->next;
            free(temp);
        }
    }
}
```

```c
void Peek(struct node *p){
    if(p->data==-100) printf("Queue Underflow!\n\n");
    else{
        printf("%d Peeked!\n\n", p->data);
    }
}


void Display(struct node *p){
    if(p->data==-100) printf("Queue is Empty!\n\n");
    else{
        while(p!=NULL){
            printf("%d\t", p->data);
            p = p->next;
        }
        printf("\n");
    }
}

int main(){
    struct node *head = malloc(sizeof(struct node));
    head->data = -100;
    while(1){
        int n;
        printf("1.Push\t2.Pop\t3.Peek\t4.Display\t5.Exit\nEnter
Your Choice: ");
        scanf("%d", &n);
        if(n==1) Push(head);
        else if(n==2) Pop(&head);
        else if(n==3) Peek(head);
        else if(n==4) Display(head);
        else if(n==5) exit(0);
        else printf("----Please Enter Correct One!-----\n\n");
    }
return 0;
}
```