→ To create node we will create structure

```
struct node
  {
     int info;
     struct node *link;
  }
```

20-sept-19

Write a program to create a link list.

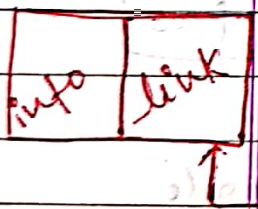
```
#include <stdlib.h>
#include <stdio.h>
#include <conio.h>
#include <alloc.h>

struct Node
  {
     int info;
     struct node *link;
  };

typedef struct Node node;

void main()
  {
     int choice;
     char ch='y';

     node *start, *ptr, *new;
```



creation of Node through structure

struct Node converted into Node

```
while (ch == 'y' || ch == 'Y')
{
    printf("enter 1 for create linked list \n");
    printf("enter 2 for display linked list \n");
    printf("enter 3 for exit");
    scanf("%d", &choice);
    switch(choice)
    {
        case 1: insert();
                break();
        case 2: display();
                break();
        case 3: exit();
        default: printf("Invalid choice");
    }
    printf("press y/Y for continue");
    scanf("%d", &ch);
}
getch(); }
```

start

keep the address of Nodes

```
void insert ()
{
    data type = node
    node *start = NULL, *ptr, *nw;
    int val;
```

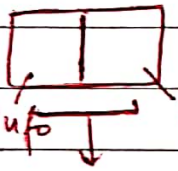
for insert, delete etc operations

```
    printf ("enter value which you want to insert in node");
    scanf ("%d", &val);
```

provide address of node to ptr or memory

data type

```
    ptr = (Node*) malloc (size of (node));
```



garbage value insid node

```
    ptr -> info = val;
    ptr -> link = NULL;
    if (start == NULL) → first node
    {
        start = ptr; → first node assigned to START
    }
    else
    {
        nw = start;
        while (nw -> link != NULL)
        {
            nw = nw -> link;
        }
        nw -> link = ptr;
    }
}
```

going to the last node where link = NULL.

PAGE NO. :
DATE : / /

```
void display ()
{
    node * temp;
    temp = start;

    while (temp != null)
    {
        printf("%d\t", temp->info);
        temp = temp->link;
    }
}
```

(datatype *) malloc (size of (name of datatype));

(float *) malloc (size of (float));
(int *) malloc (size of (int));

memory by → Malloc → contain garbage value
→ Calloc → index start with zero, one, two...

Malloc contains garbage value

Calloc is index wise starts with zero then one then two and so on.

```
Void delete()
{
    node *ptr, *nw;
    int val, del = 1;

    printf("enter the number of element
            you want to delete");

    scanf("%d", &val);
    nw = start;

    while (del != val+1)
    {
        yw = nw -> link;
        a = nw -> link;

        del++;
    }
    del = 1;   nw = start;

    while (del != val)
    {
        nw = nw -> link;
        if (del == val-1)
        {
            nw -> link = a;
        }
        del++;
    }
}
```

```
void insert_beg (start)
{
    a = (node*) malloc( sizeof (node));
    printf ("enter value ");
    scanf ("%d", &val);
    a -> info = val;
    a -> link = null;
    ~~a -> link~~
    ~~start~~
    ptr -> link = start;
```

```
    start = a;
    free(a);
}

void delete (start)
{
    node *temp;
    temp = start;
    start = start -> link;
    temp -> link = null;
    printf ("deleted value is = %d", temp
                                    -> info);
    free (temp);
}

Insertion_after_value (start)
{
    int val, val2, flag = 0;
    printf (" Enter value which you want
                to insert ");
    scanf (" %d ", &val1);
    nw = (node*) malloc (sizeof (node));
    nw -> info = val;
    nw -> link = NULL;
    printf (" enter value ");
    scanf ("%d", &val2);
    ptr = start;
```

```
while (ptr != NULL)
{
    if (ptr->info == val(2)
    {
        flag = 1;
        break;
    }
    ptr = ptr->link;
}
if (flag == 0)
{
    printf ("cannot insert");
    exit();
}
else
{
    nw->link = ptr->link;
    ptr->link = nw;
}
}
```

use my study
material at
your own risk