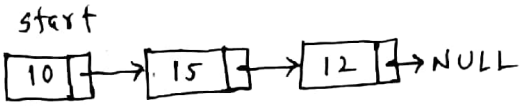
Linkedlist (Singular or Linear)

Write a program to create a linear linkedlist in memory and to display its contents.

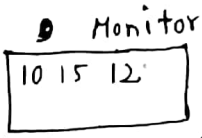
Part-I

Part-II

Part-I:



Part-II;



Ans:

```
#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>
 struct list {
   int data;
   struct list *next;
  };
 typedef struct list node;
 void display(node *head);
int main() {
  node *start, *prev, *temp;
  char ans;
  start = NULL;
  do{ fflush(stdout);
     printf("Do you want to create(Y|N):=");
     ans = toupper(getche());
     if(ans=='Y'){
        if(start==NULL){
          start = (node*) malloc(sizeof(node));
          scanf("%d", &start->data);
          start->next = NULL;
          prev = start;
        }
        else {
          temp = (node*) malloc(sizeof(node));
          scanf("%d", &temp->data);
          temp->next = NULL;
          prev->next = temp;
          prev = temp;
        }
        printf("\n");
     }
  } while(ans!='N')
  display(start);
  return 0;
}
```

printf("\n Enter Data:=");

printf("\n Enter Data:=");

```
void display(node *head){
   node *temp;
   temp = head;
   while(temp!=NULL){
    printf("%d ", temp->data);
    temp = temp->next;
   }
   return;
}
```

printf("created list:=");

stdlib.h → malloc(), ctype.h → toupper()

struct list {—} → data | next

node *start;

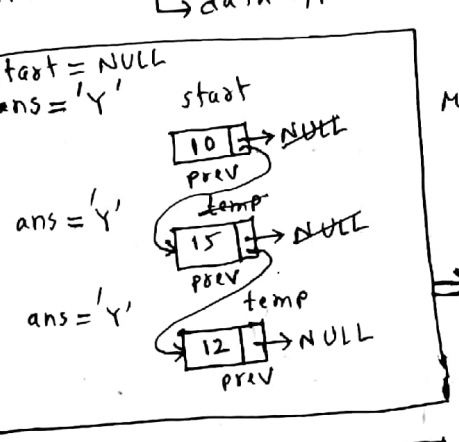
Address
Variable name

int x → variable

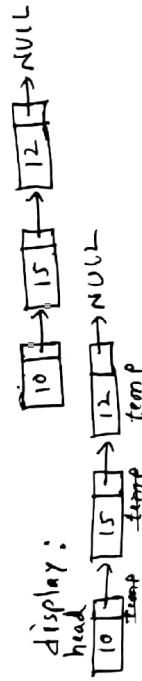
int → data type

node → data type

start = NULL
ans = 'Y'



Memory



After calling display: head

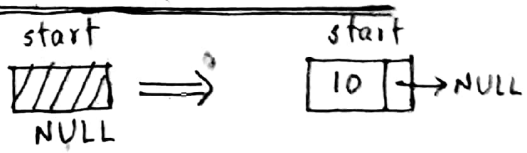
Monitor

```
Do you want to create(Y|N):=Y
Enter Data:=10
Do you want to create(Y/N):=Y
Enter Data:=15
Do you want to create(Y/N):=Y
Enter Data:=12
Do you want to create(Y/N):=N
created list:=10 15 12
```

if(True) ⇒ if(True) ⇒ while(True)
if(True) ⇒ if(False) ⇒ else ⇒ while(True)
if(True) ⇒ if(False) ⇒ else ⇒ while(True)
if(False) ⇒ while(False) ⇒ display()

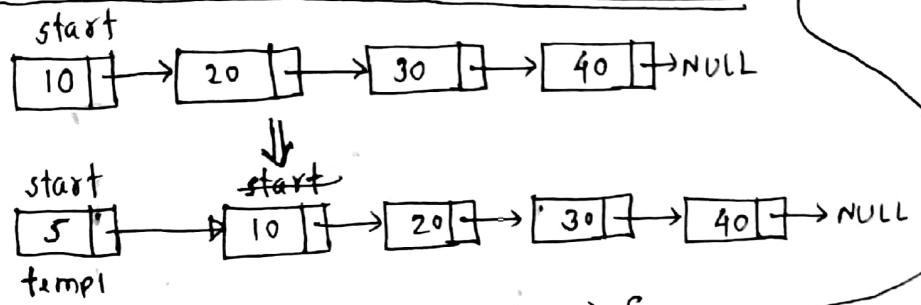
Insertion in a linear linkedlist

start = NULL (newitem = 10)



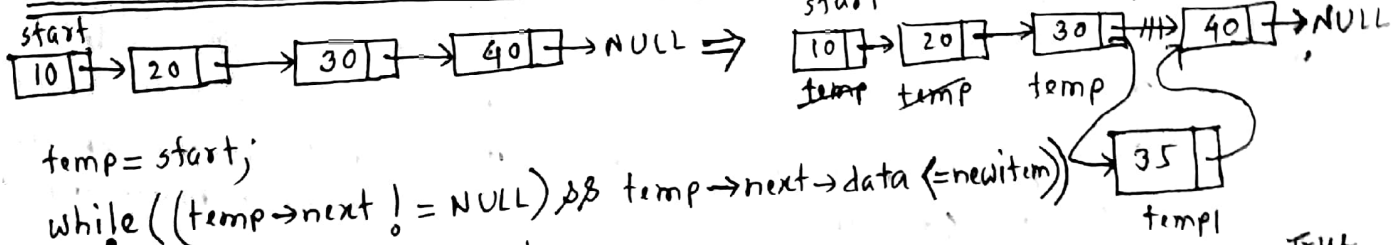

```
if (start == NULL) {
    start = (node*) malloc (sizeof (node));
    start → data = newitem;
    start → next = NULL;
}
```

(newitem = 5) First Insertion




```
if (start → data >= newitem) {
    temp1 = (node*) malloc (sizeof (node));
    temp1 → data = newitem;
    temp1 → next = start;
    start = temp1;
}
```

Middle or last Insertion (newitem = 35)




```
temp = start;
while ((temp → next != NULL) && temp → next → data <= newitem)
{
    temp = temp → next;
}
temp1 = (node*) malloc (sizeof (node));
temp1 → data = newitem;
temp1 → next = temp → next;
temp → next = temp1;
```

True && 20 <= 35 = True (⇒ True)
True && 30 <= 35 = True
True && 40 <= 35 = False (40 <= 35 : false)

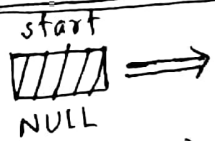
Insertion logic

```
if (start == NULL) {
    ≡
}
else if (start → data >= newitem) {
    ≡
}
else {
    Middle / Last insertion logic
}
```

```
node *Insertion (node *start, int newitem)
{
    node *temp, *temp1;
    // Insertion Logic
    return start;
}
```

Deletion from a linear linked list

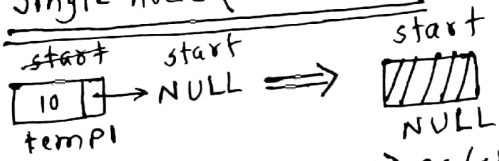
start = NULL (item = 10)



⟹ Message "No linkedlist"

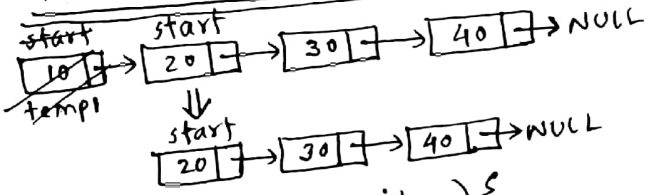
```
if (start == NULL) {
    printf("\n No Linked list");
}
```

Single node (item = 10)



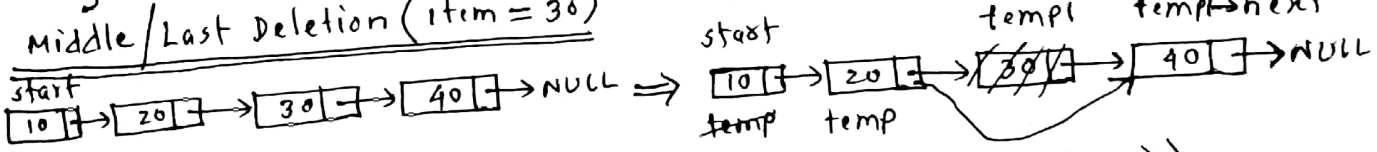

```
if (start->next == NULL) && (start->data == item))
{
    temp1 = start;
    start = start->next;
    free (temp1);
}
```

First Deletion (item = 10)




```
if (start->data == item){
    temp1 = start;
    start = start->next;
    free (temp1);
}
```

Middle / Last Deletion (item = 30)




```
temp = start;
while ((temp->next != NULL) && (temp->next->data != item))
{   temp = temp->next;
}
temp1 = temp->next;
temp->next = temp1->next;
free(temp1);
```

True && 20 != 30 = True

True && 30 != 30 = False

(True, False)

Deletion Logic

```
if (start == NULL) {

}
else if (start->data == item) {

}
else {
    Middle/Last Deletion Logic
}
```

```
node *Deletion (node *start, int item){
    node *temp, *temp1;
    // Deletion Logic
    return start;
}
```