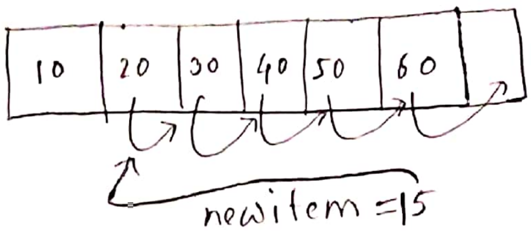
Linked list:

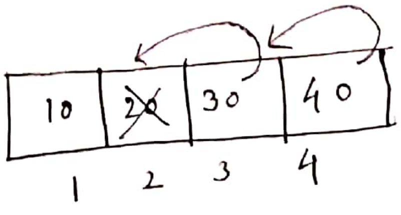
Why linkedlist is needed?

Array

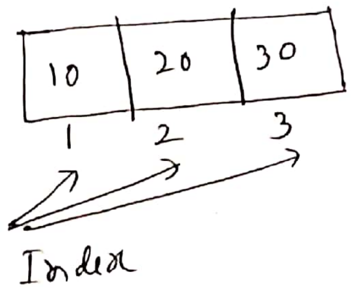
1. Insertion difficult



2. Deletion difficult

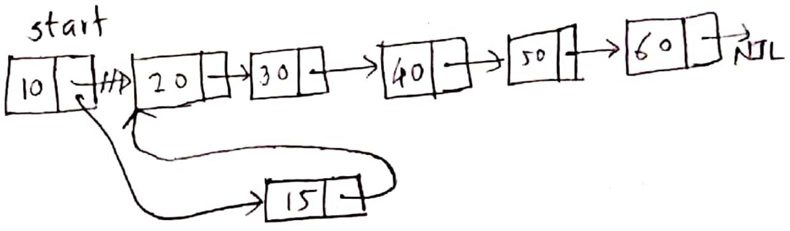


3. Binary search easy

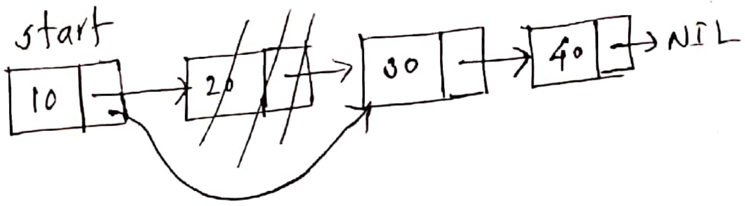


Linkedlist

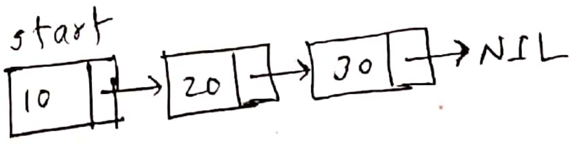
1. Insertion easy



2. Deletion easy

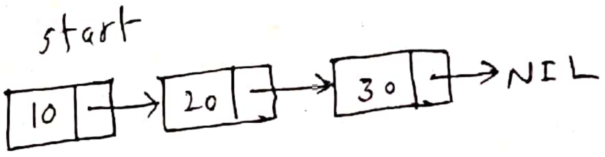


3. Binary search difficult

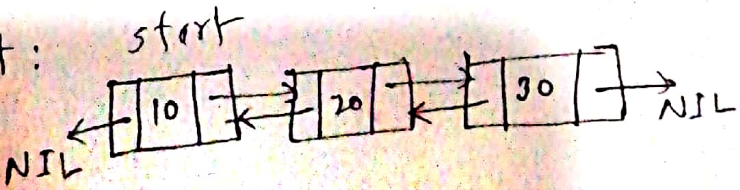


different linkedlists

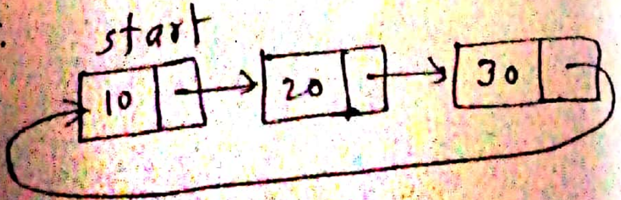
Linear or singular linkedlist:



Doubly or Two-way linkedlist:



Linear circular linkedlist:



Memory Allocation in linkedlist:

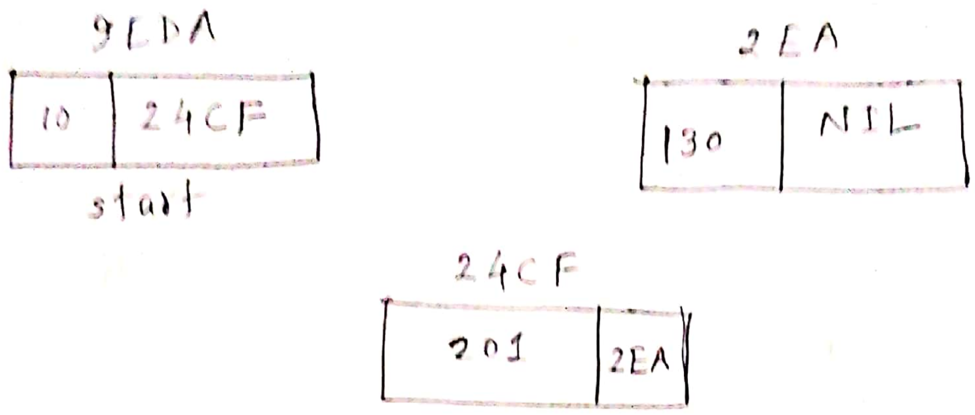
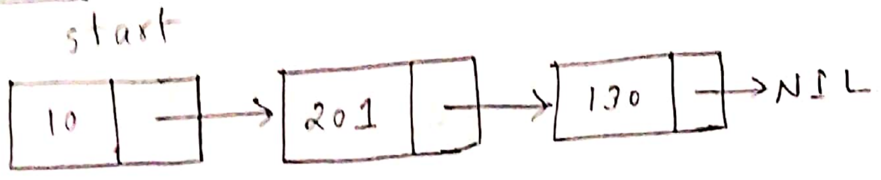
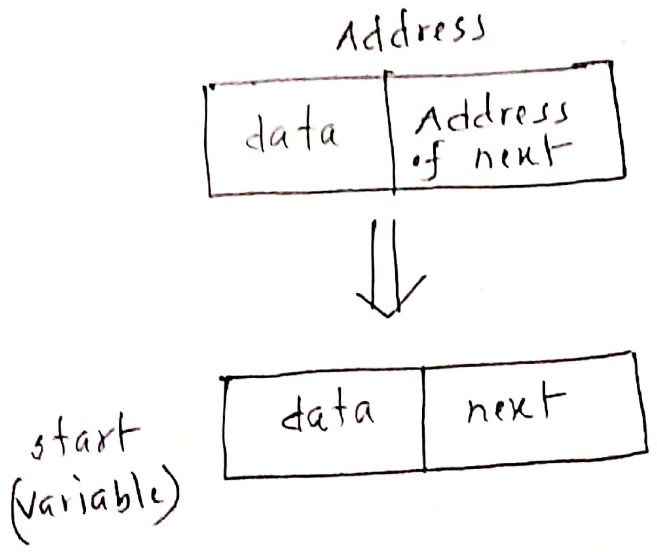


Diagram:



Each node

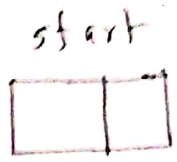


Variable Declaration:

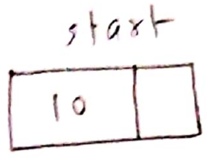
```
struct list {
        int data;
        struct list *next;
    };
typedef struct list node;
int main() { node *start;
           }
```

~~creation of linkedlist~~ statements

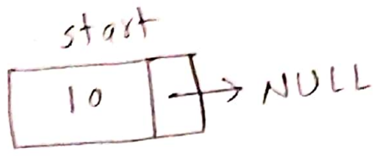
1. start = (node*)malloc(sizeof(node));



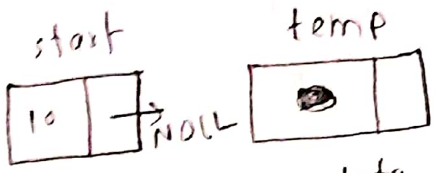
2. start→data = 10;



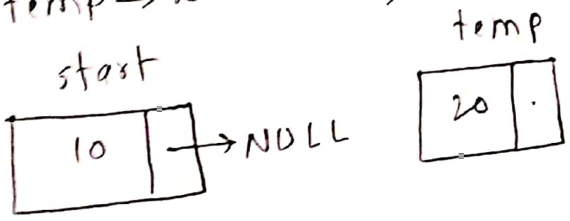
3. start→next = NULL;



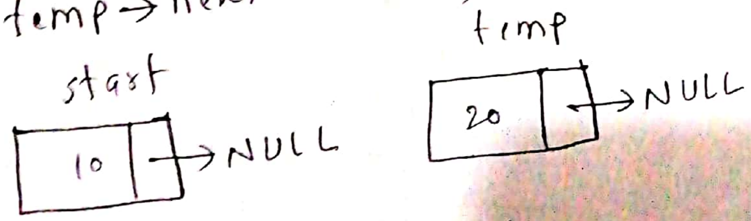
4. temp = (node*)malloc(sizeof(node));



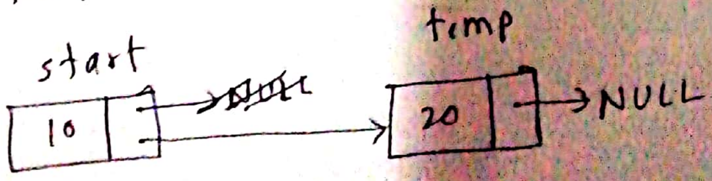
5. temp→data = 20;



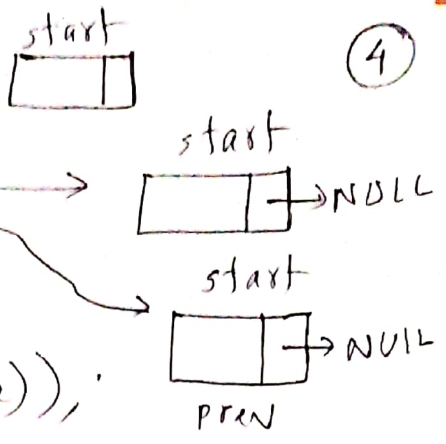
6. temp→next = NULL;



7. start→next = temp;



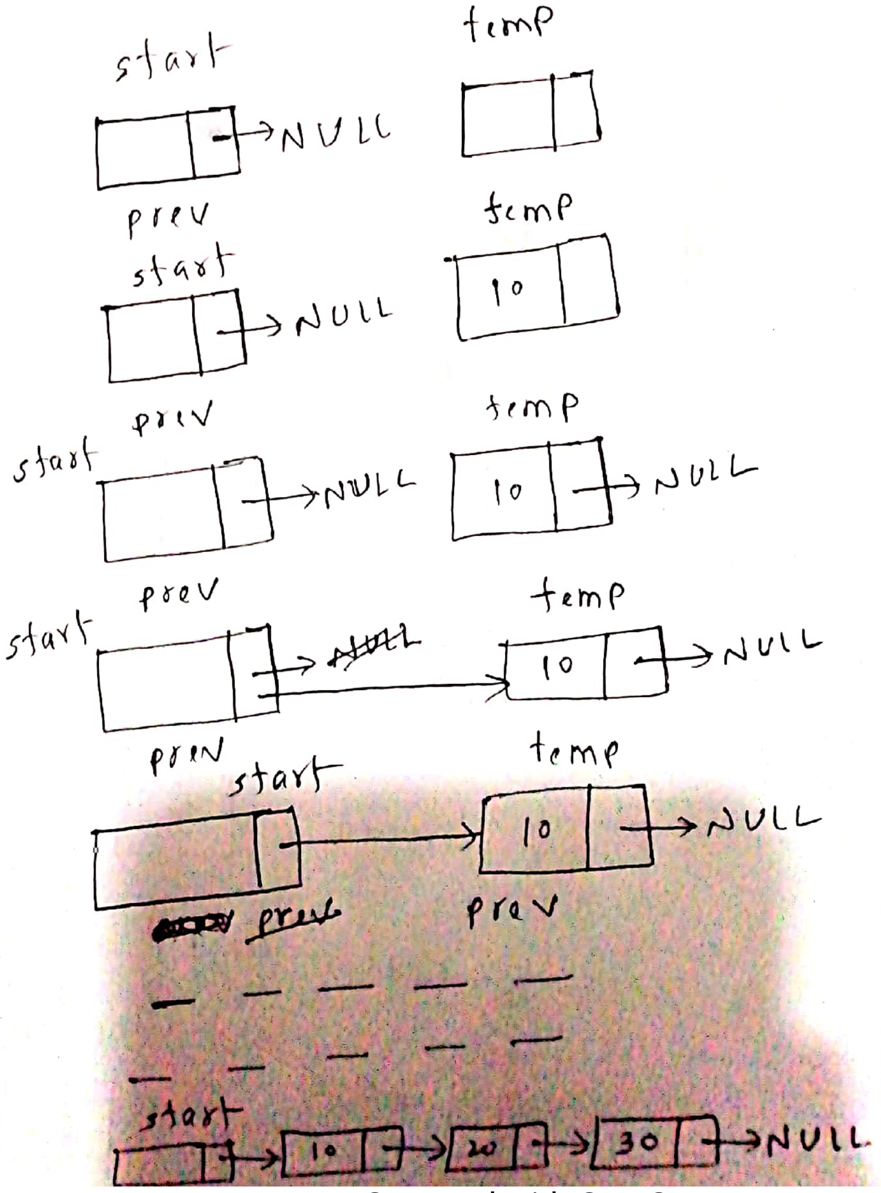
```
start = (node*)malloc(sizeof(node));
start->next = NULL;
prev = start;
for(i = 10;  i<=30;   i= i+10){
                True
    temp = (node*)malloc(sizeof(node));
    temp->data = i;
    temp->next = NULL
    prev->next = temp;
    prev = temp;
}
```



i = 10   i <= 30
  ⇒ 10 <= 30 True

i = 20   i <= 30 True

i = 30   i <= 30 True


start
temp
NULL
prev
start
temp
10
NULL
start
prev
temp
10
NULL
NULL
start
prev
temp
~~NULL~~
10
NULL
prev
start
temp
10
NULL
prev
prev
start
10
20
30
NULL