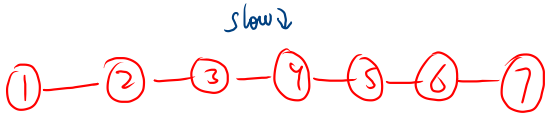
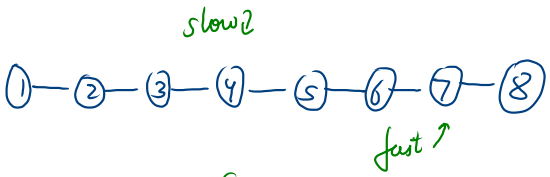
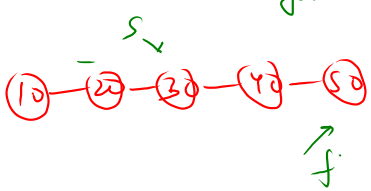


while (fast.next != null)
movement

fast ↑ while (fast.next != null && fast.n.n != null)





W ( ) {
s
f
}

dis = speed * time

itr 1 → dis1 ⇒ xt

itr 2 → dis 2 ⇒ 2xt

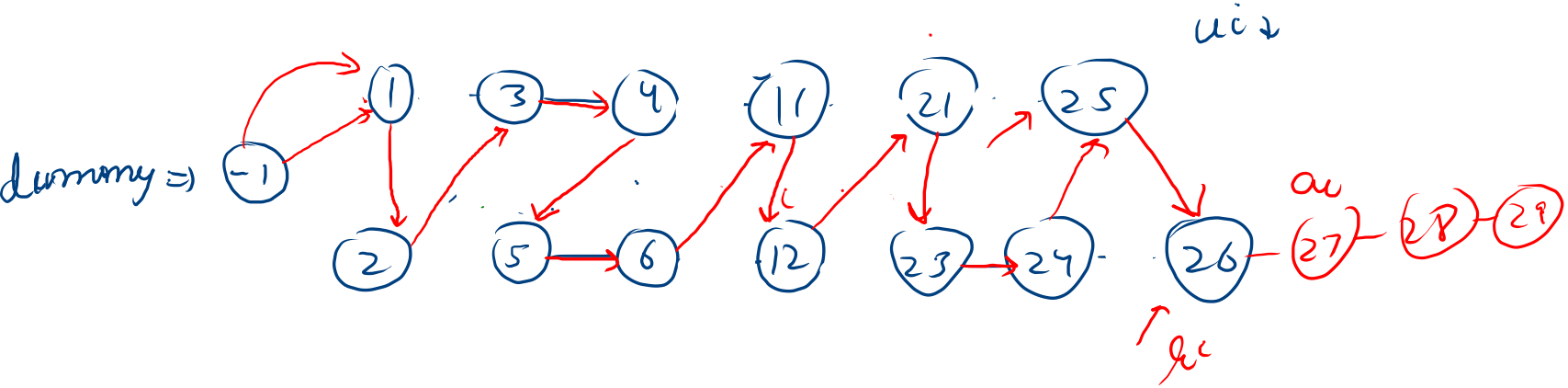
2xt = size

$xt = \frac{size}{2}$

xt ⇒ dis1

$dis1 \Rightarrow \frac{size}{2}$

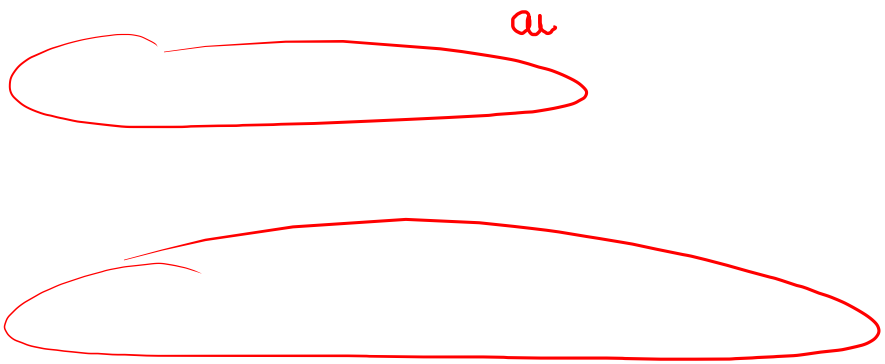
$itr1 \Rightarrow \frac{size}{2}$



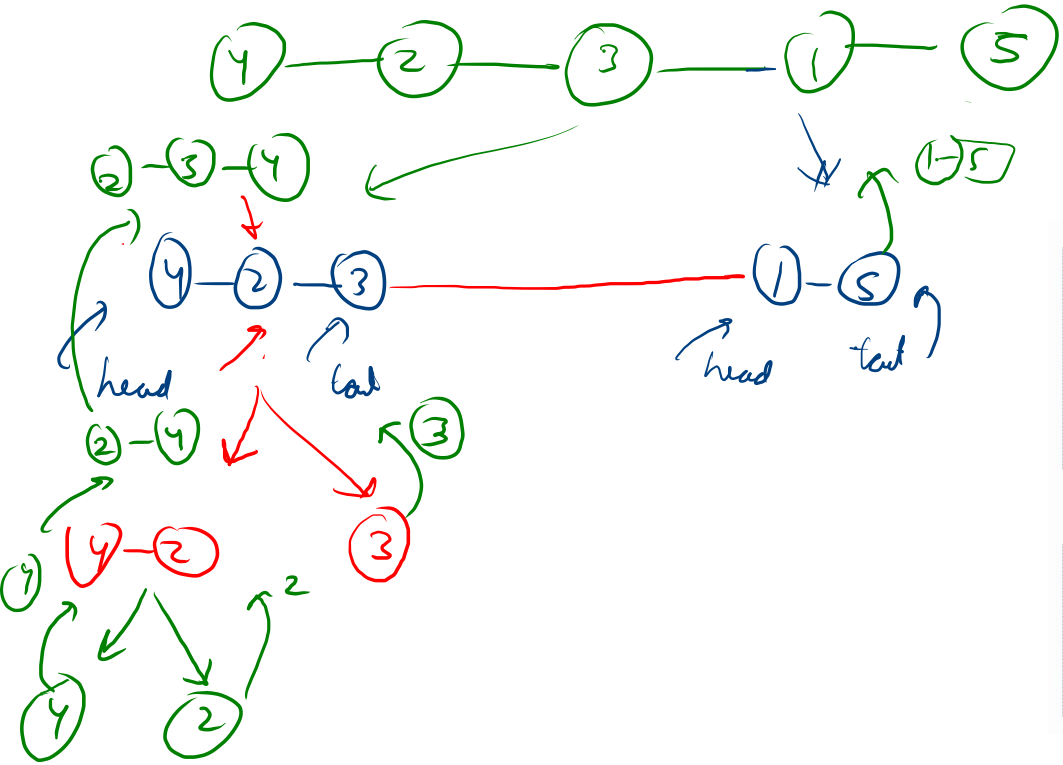
ai.next = ui

ai.next = li

return dummy.next.




```
LinkedList ans=new LinkedList();

Node ui=l1.head;
Node li=l2.head;

while(ui!=null && li!=null){
    if(ui.data<li.data){
        ans.addLast(ui.data);
        ui=ui.next;
    } else {
        ans.addLast(li.data);
        li=li.next;
    }
}

while(ui!=null){
    ans.addLast(ui.data);
    ui=ui.next;
}

while(li!=null){
    ans.addLast(li.data);
    li=li.next;
}

return ans;
```



→ f.n != null
↓
→ f != tail

f.n.n != null
↓
f.n != tail

```
public static Node getmid(Node head,Node tail){
    Node slow=head;
    Node fast=head;

    while(fast!=tail && fast.next!=tail){
        slow=slow.next;
        fast=fast.next.next;
    }

    return slow;
}

public static LinkedList mergeSort(Node head, Node tail){

  Node mid=getMid(head,tail);

  LinkedList l1=mergeSort(head,mid);
  LinkedList l2=mergeSort(mid.next,tail);

  return mergeTwoSortedLists(l1,l2);
}
```

959

1632 1569
928 947
827 ←
839
1970
1579

O/1 knapsack
1789
765

DFS

1192, 1706
417 841
743
1466
1600

Course schedules / Job scheduling

Topo
↳ 802

→ clone a graph

Minimum number of semesters

Topo

269
329
2050

Course schedule 1, 2, 3, 4

Medium 1344, 1386

1877 1753 881
921 1011 * 945
969 1481 2126
1963 11 435

Greedy

44 517 1713 936
871 1671 410 1665
2141 1326 2136