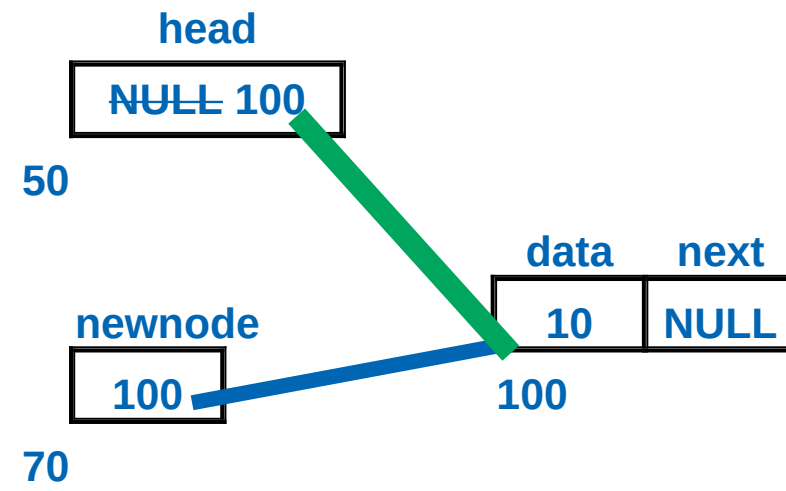**Case 1 : add first when linked list is empty**

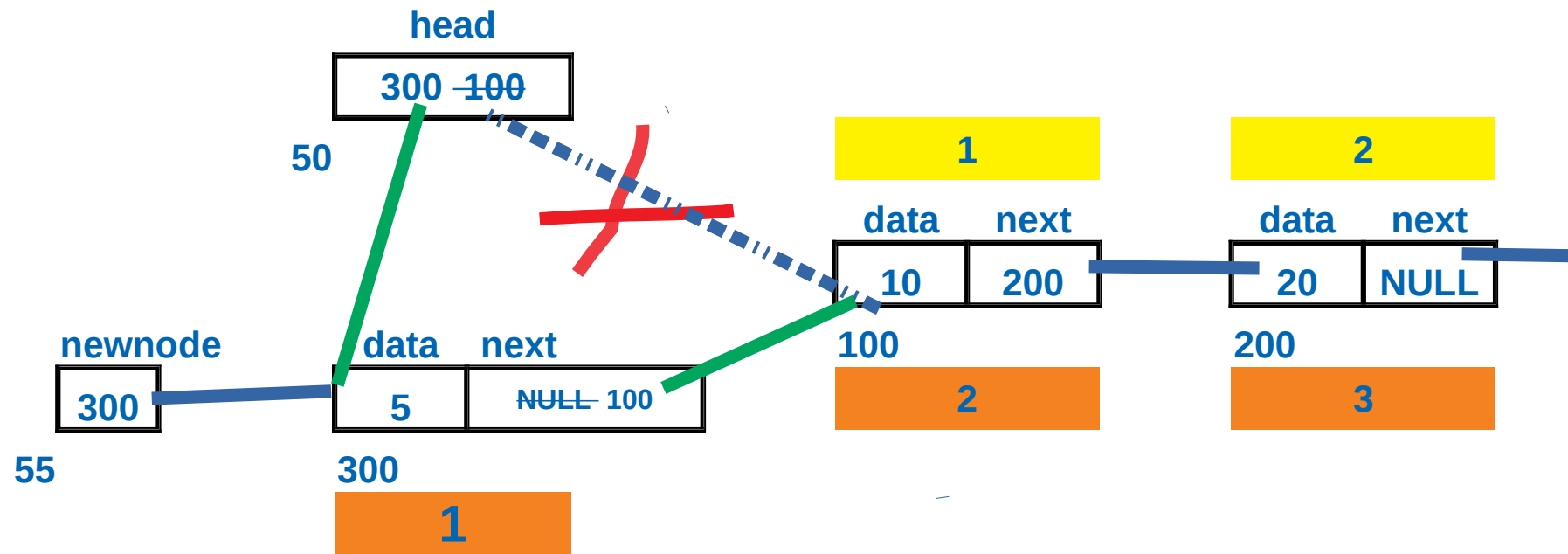**void AddFirst(int value)**
{
   node_t *newnode=NULL;
   newnode= CreateNode(value);
   if(head==NULL)
   head=newnode;
   else
   {
      newnode->next=head;
      head=newnode;
   }
   return ;
}

head

NULL **100**

50

Addfrist(10);

data **next**

newnode **10** **NULL**

100 **100**

70

Case 2 : add first when list is has multiple nodes

head

300 **100**

50

1 **2**

data **next** **data** **next**

10 **200** **20** **NULL**

newnode **data** **next** **100** **200**

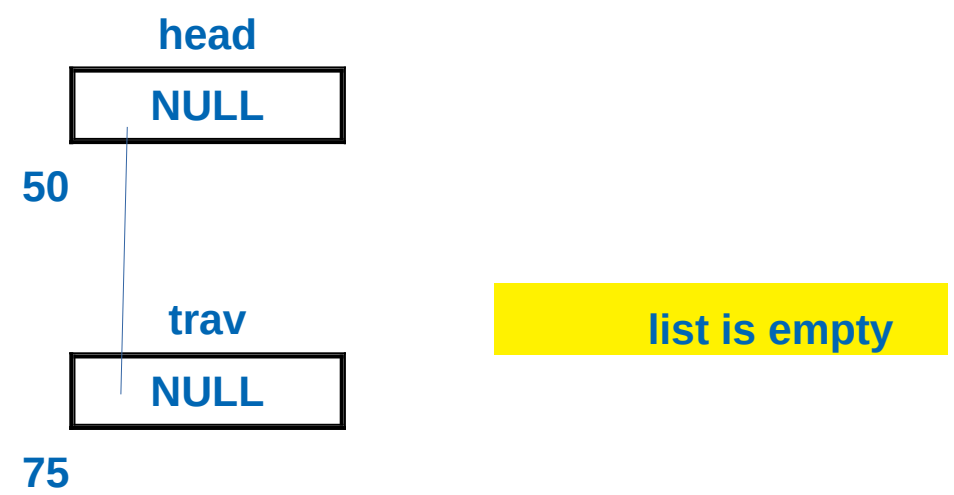**300** **5** **NULL** **100** **2** **3**

55

300

1

Case 1 : display when list is empty
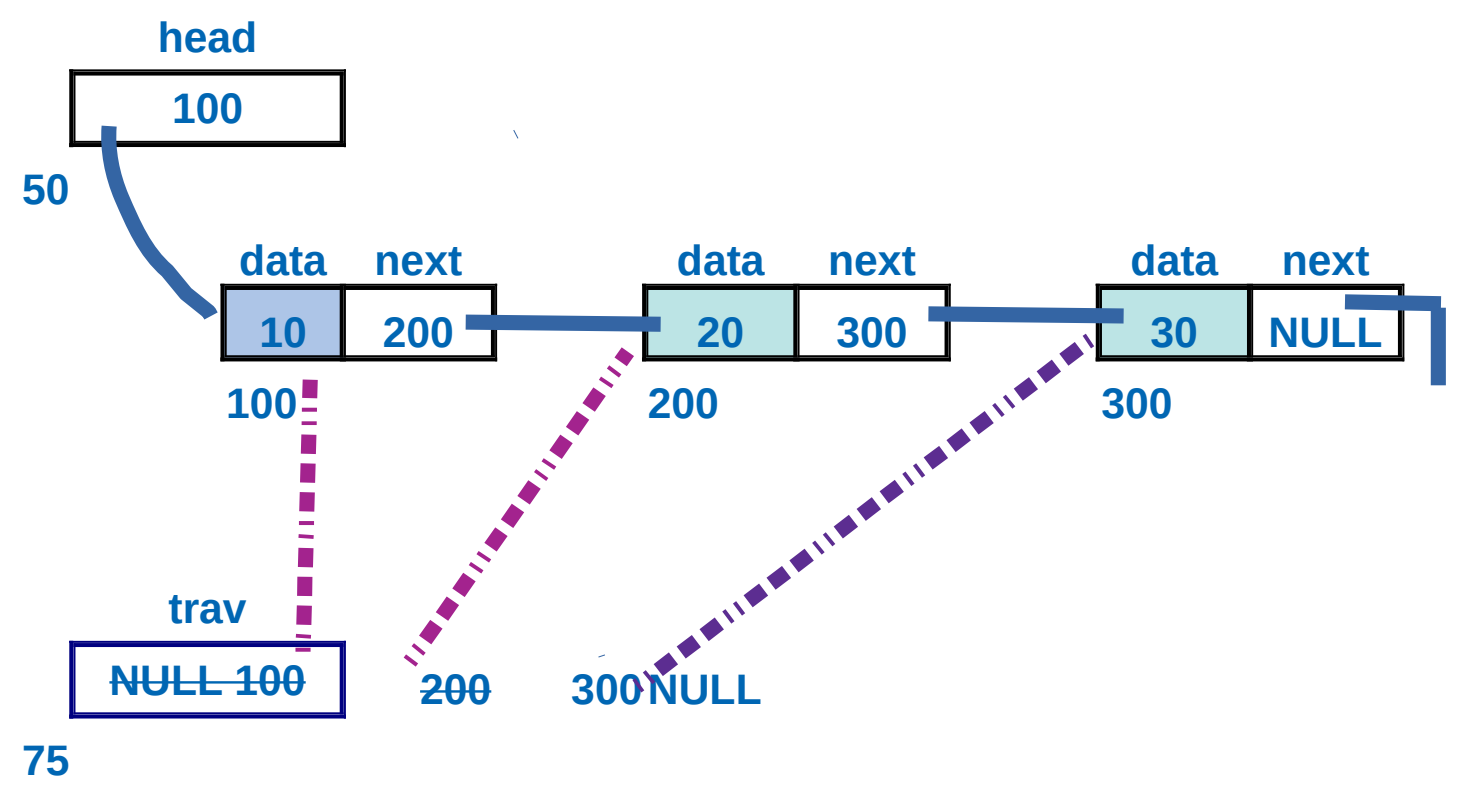
```
void DisplayList()
{
    node_t * trav=NULL;
    trav=head;
    if(trav==NULL)
    {
    printf("\n list is empty");
    }
    else
    {
        while(trav!=NULL)
        {
        printf("%d--->", trav->data);
        trav=trav->next;
        }
    }
    printf("\n");
    return;
}
```

head

NULL

50

trav

NULL

75

list is empty

Case 2 : display when list has multiple nodes

head

100

50

data	next
10	200

100

data	next
20	300

200

data	next
30	NULL

300

10   20   30

trav

NULL 100

75

200   **300 NULL**

while(100!=NULL)   TRUE

While(200!=NULL)   TRUE

While(300!=NULL)   TRUE

While(NULL!=NULL)   FALSE

```
void AddLast(int value)
{
    node_t *newnode=NULL;
    newnode=CreateNode(value);
    if(head==NULL)
        head=newnode;
    else
    {
        NODE *trav=head;
        while(trav->next!=NULL)
        {
            trav=trav->next;
        }
        trav->next=newnode;
    }
    return ;
}
```

Case 1 : add last when list is empty

head

NULL 100

50

	data	next
	10	NULL

newnode

100

60

100
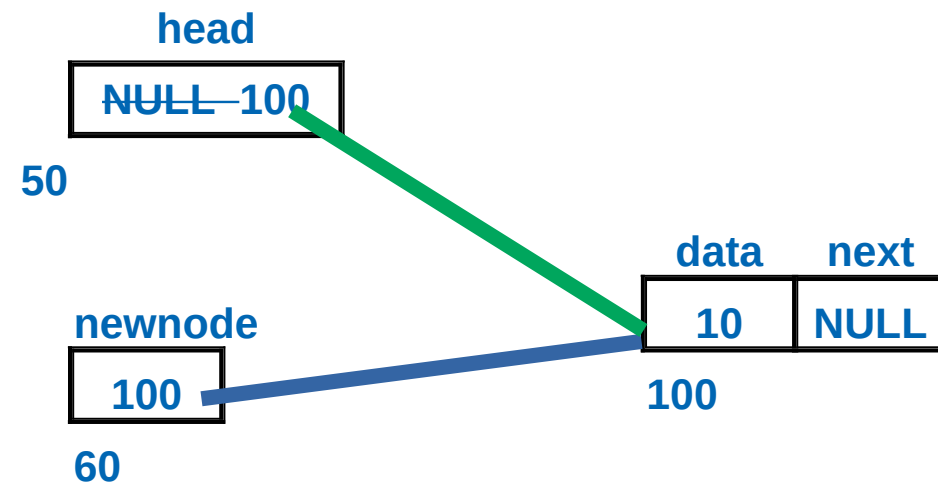
Case 2 : add last when list is has multiple nodes

head

100

50

1

data	next
10	200

100

2

data	next
20	300

200

3

data	next
30	400

300

4

data	next
40	NULL 500

400

5

data	next
50	NULL

500

trav

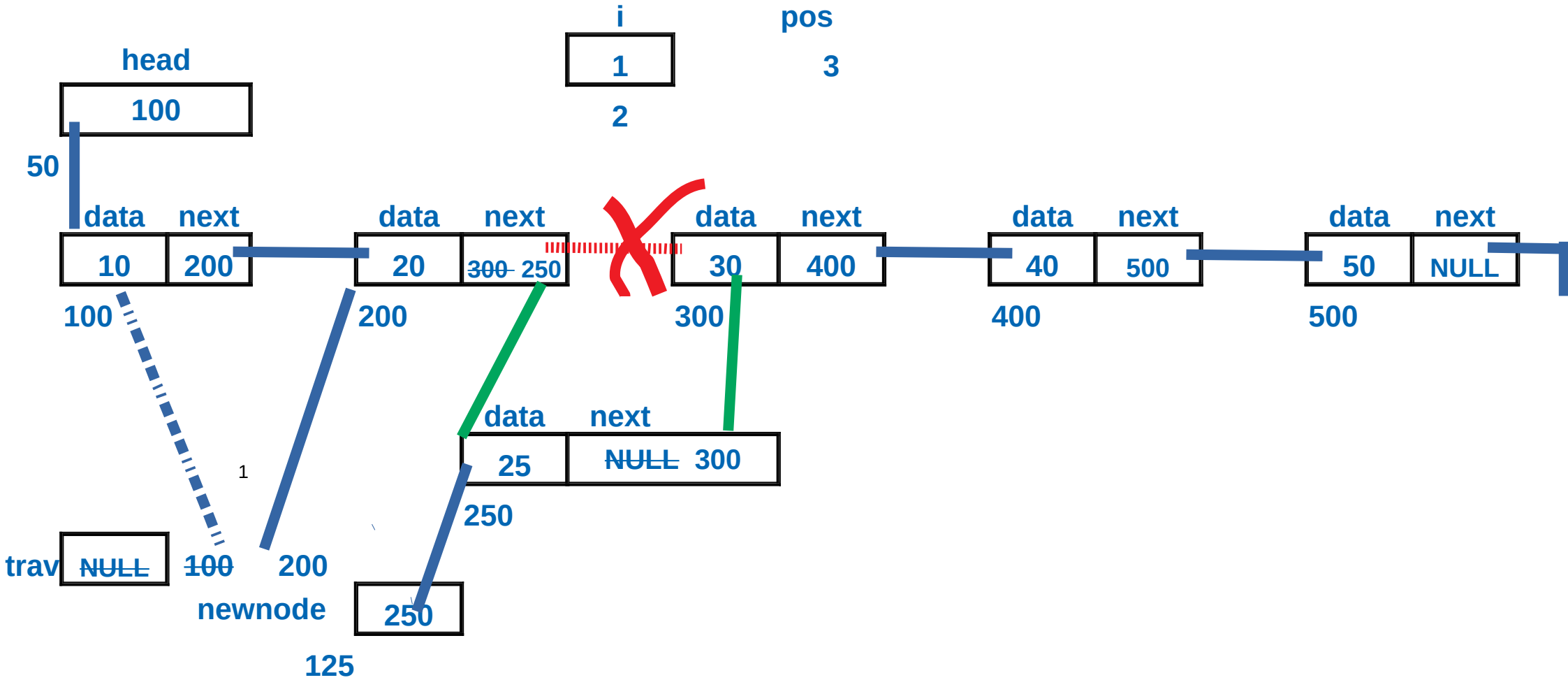| NULL	100	200	300	400

newnode

500

80
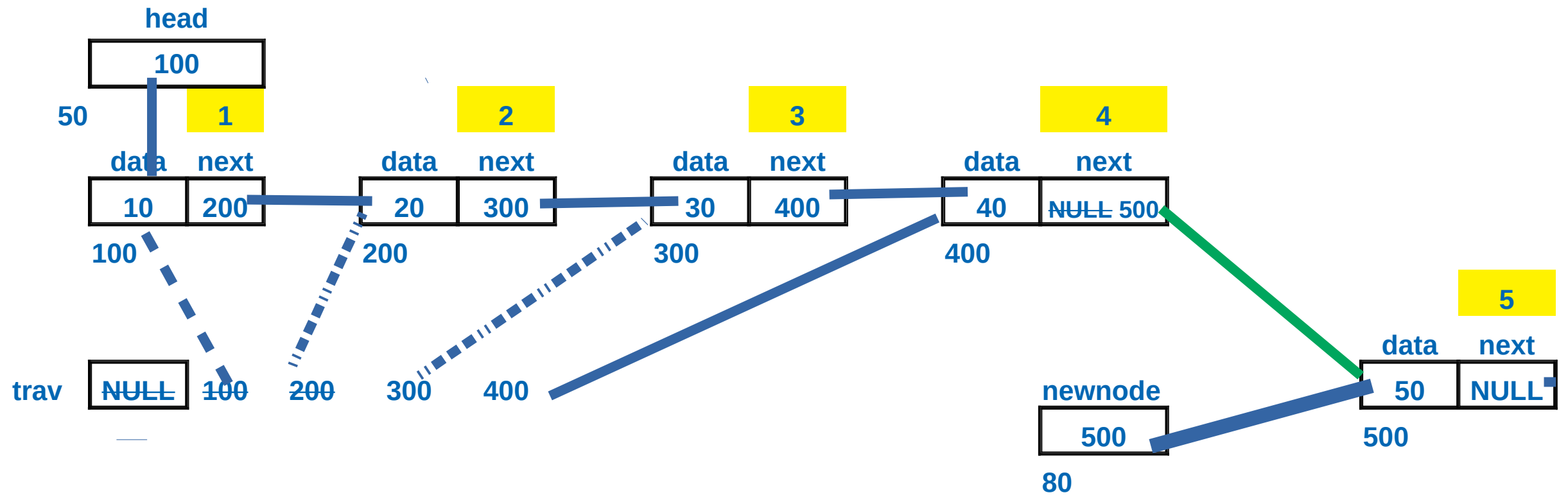
While(200!=NULL)   TRUE

While(300!=NULL)   TRUE

While(400!=NULL)   TRUE

While(NULL!=NULL)   FALSE

Case 1 : add at 3 rd position when list is has 5 nodes

```
void AddAtPosition(int position, int value)
{
node_t *newnode=NULL;
node_t *trav=NULL;
int i;
if(position==1)
AddFirst(value);
else if( (CountNodes()+1)==position)
AddLast(value);
else
{
newnode= CreateNode(value);
trav=head;
for(i=1; i<position-1; i++)
{
trav=trav->next;
}
newnode->next=trav->next;
trav->next=newnode;
}
return;
}
```

i

1
2

pos

3

head

100

50

data	next
10	200

100

data	next
20	300 250

200

data	next
30	400

300

data	next
40	500

400

data	next
50	NULL

500

data	next
25	NULL 300

250

trav | NULL | 100 | 200

newnode

250

125

1

Case 1 : Print Linked List In Reverse

head

100

50

```
void trav_rev(node_t * trav)
{
  if(trav== NULL)
     return;
  else
     trav_rev(trav->next);
  printf("%5d-->", trav->data);
}
```

data	next
10	200

100

data	next
20	300

200

data	next
30	400

300

data	next
40	500

400

data	next
50	NULL

500

trav

5

500