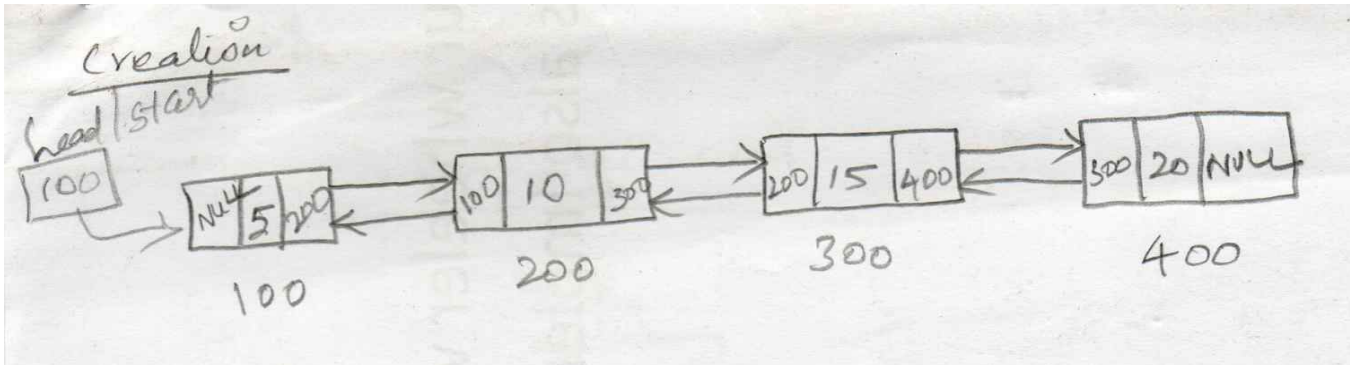
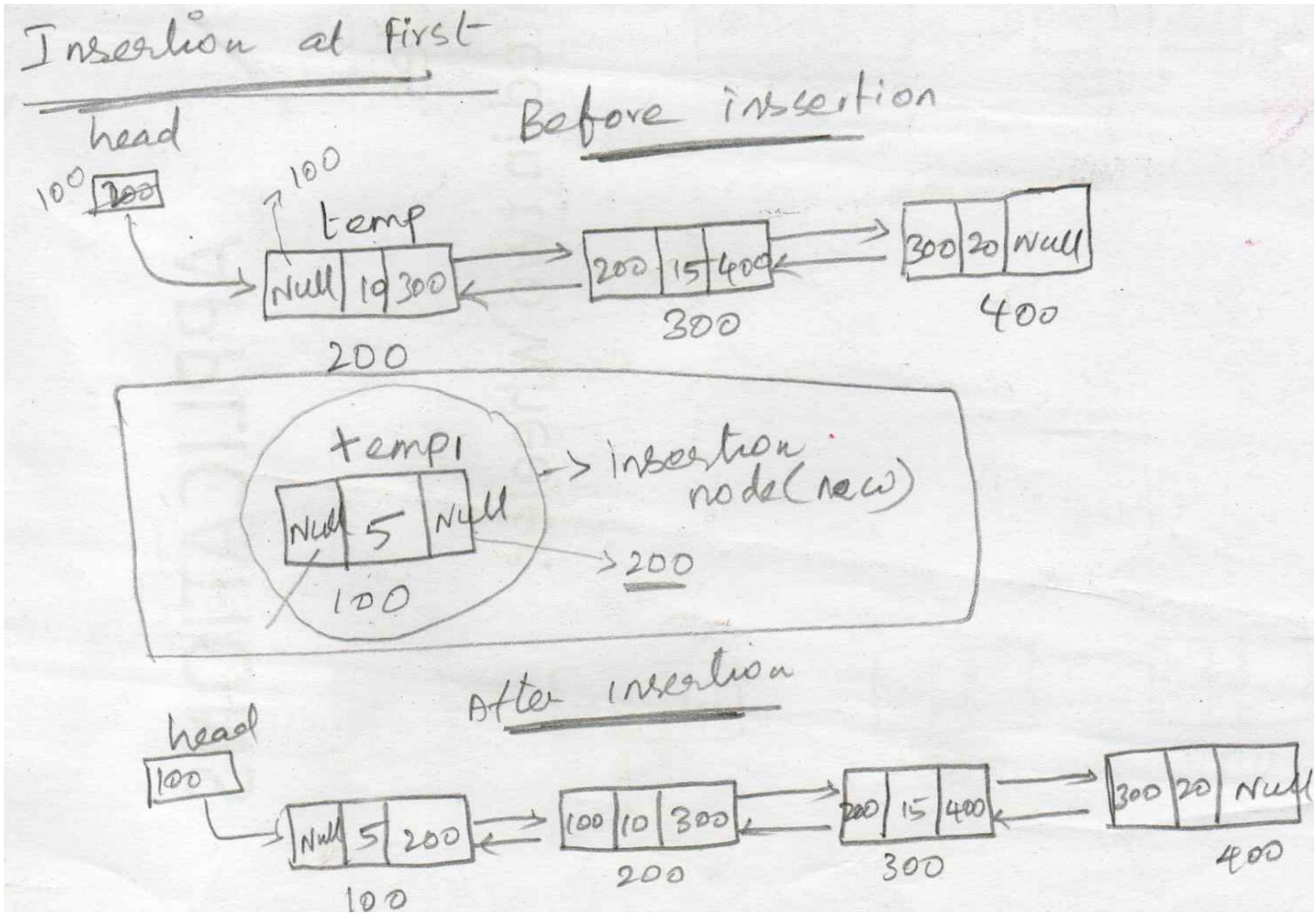
Creation of Double Linked List

```
void create()

{

*node temp, temp1;

if(head==null)

{

temp=(struct node *)malloc (size of (struct node));
printf("enter the elements");
scanf("%d",&temp->data);
temp->next=null
temp->prev=null
head=temp;
}
else
{
temp1=(struct node *)malloc (size of (struct node));
printf("enter the elements");
scanf("%d",&temp1->data);
temp->next=temp1;
temp1->next=null;
temp1->prev=temp
temp=temp1;
}
}
```



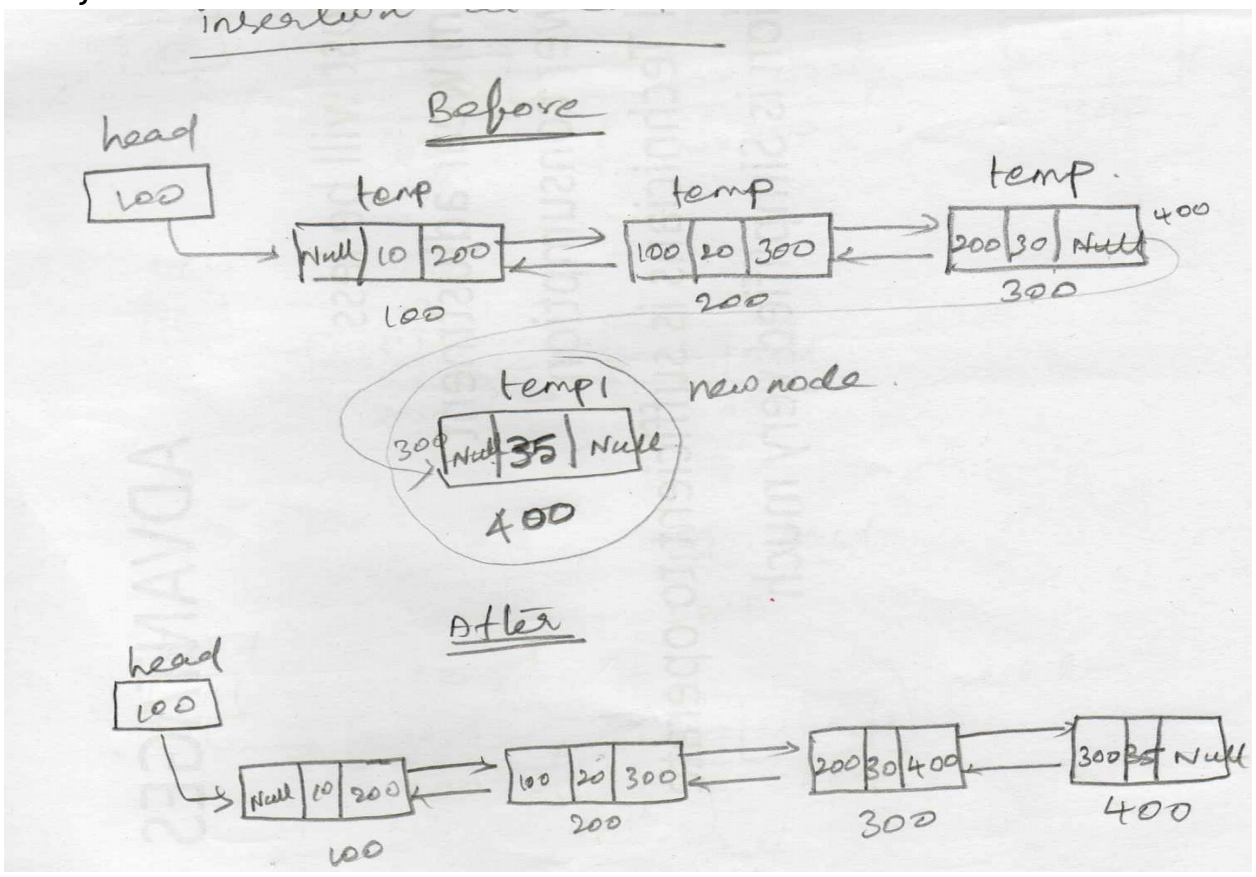
Insert node at first:

```
void insertfirst(int)

{
struct node *temp,temp1;
temp=head;
temp1=(struct node *)malloc (size of (struct node));
printf("enter the elements");
scanf("%d",&temp1->data);
 temp1->prev=null;          // set previous address field of new node is NULL
 temp1->next=temp;   // next address of new node is linking with starting
                                node
temp->prev=temp1      // previous address of starting node is linking with new
                              node
temp1=head;              // set the new node as starting node

}
```



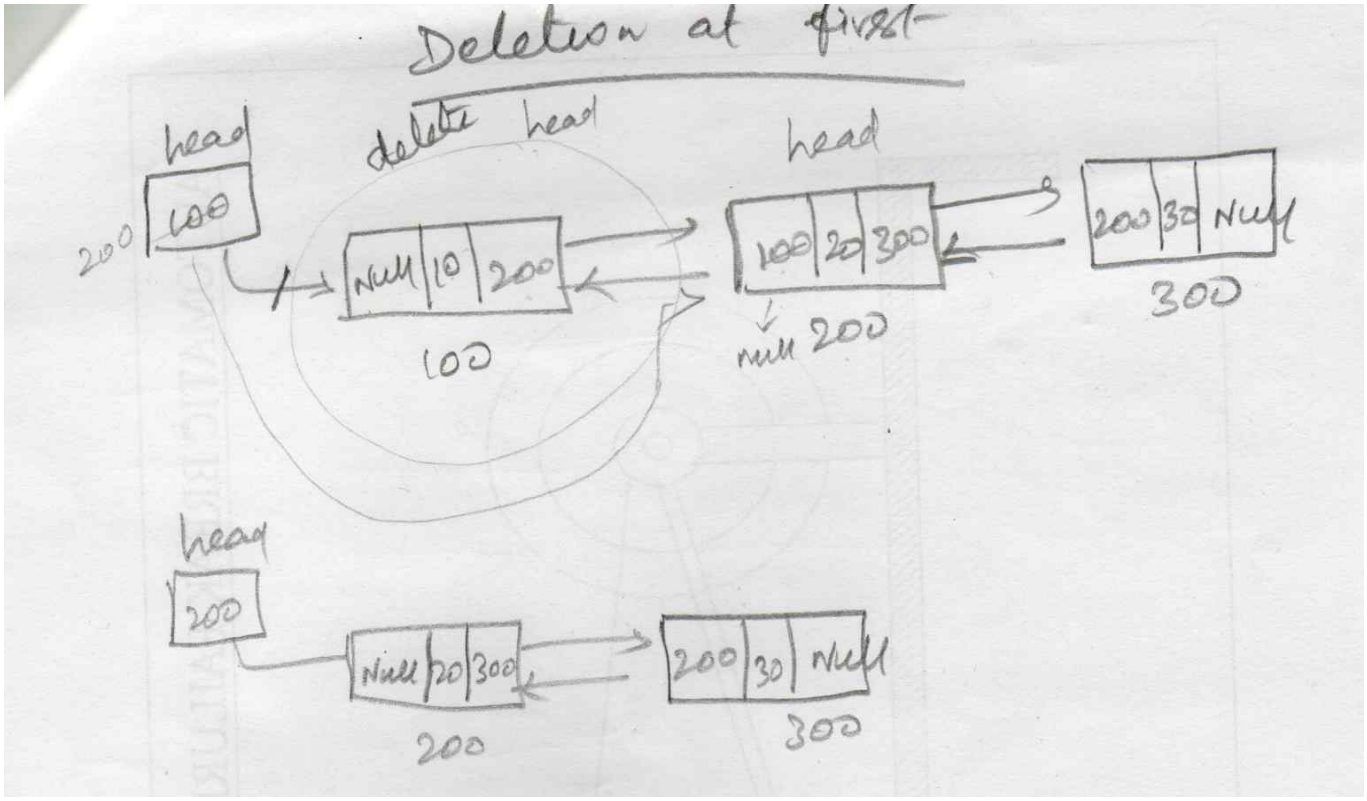
insertion at end

```
void insertend(int)

{
temp=head;
struct node *temp,temp1;
temp1=(struct node *)malloc (size of (struct node));
printf("enter the elements");
scanf("%d",&temp1->data);
 temp1->next=NULL;
temp1->prev=null;
while (temp->next! == NULL)
    {
       temp=temp->next;
     }
   temp->next=temp1; // next address of ending node is linking with new node
   temp1->prev=temp; // previous address of new node is linking with ending
                                node
      }
```



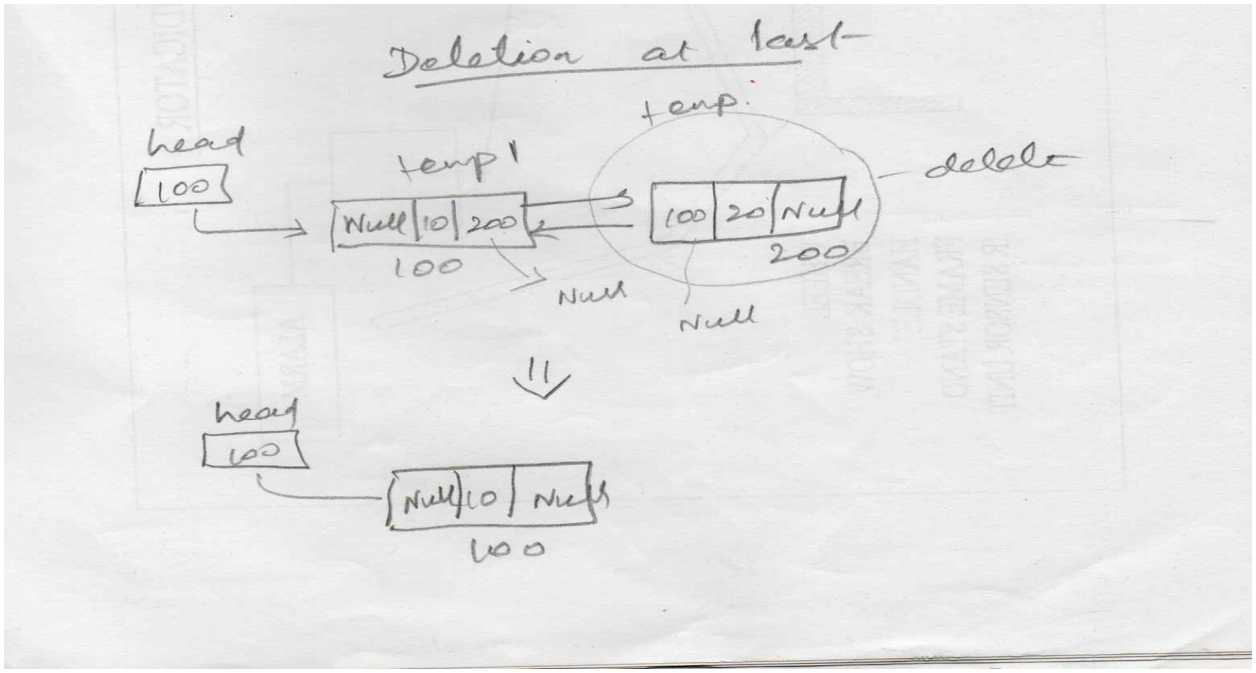
Deletion at first

```
void deletefirst()
  {
     struct node *temp;
      if(head == NULL)
     {
        printf("\nList is empty");
     }
     else
     {
       temp=head;
       head = head->next; // move the next address of starting node to 2 node
       head->prev=null  // set previous address of staring node is NULL
       free(temp);          // delete the first node from memory
      printf("\n Node deleted from the begining ...");
     }
  }
```

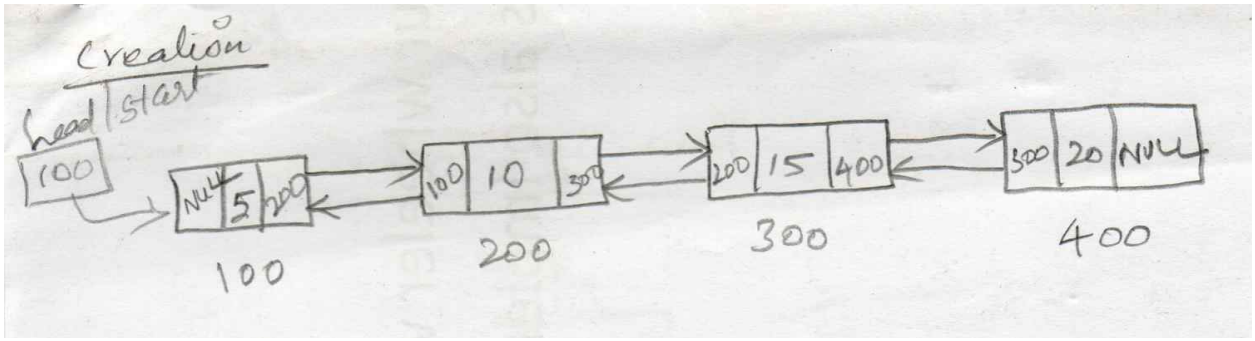


Deletion at last

```
void deletelast()
{
    struct node *temp,temp1;
     if(head == NULL)
      {
         printf("\nList is empty");
      }
      else
      {
        temp=head;
        while(temp->next!=null)
        {
        temp=temp->next;
        temp1=temp
        }
        temp1->next=null;
        temp->prev=null;
        free(temp)
}
```

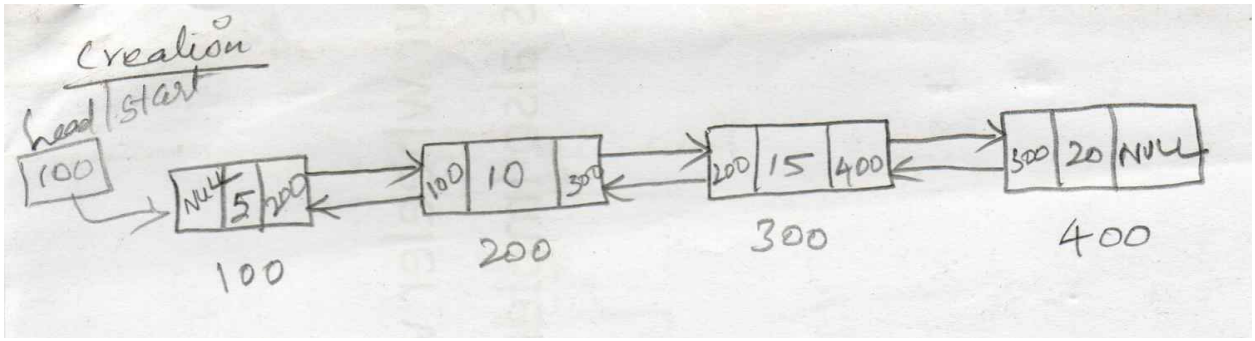


```
void displayforward()
{
   struct node * temp;

   if(head == NULL)
   {
      printf("List is empty.\n");
   }
   else
   {
      temp = head;
      while(temp != NULL)
      {
         printf("DATA ->%d, temp->data);
         temp = temp->next;
      }
   }
}
```



output 5 10 15 20

```
void reversedisplay()
{
   struct node *temp, *temp1;
   if(head == NULL)
   {
      printf("List is empty.\n");
   }
   else
   {
   temp = head;
   while(temp != NULL)
   {
      temp = temp ->next;
}
      temp1 = temp;
       printf("DATA ->%d, temp1->data);
       temp1 = temp1->prev;
}
```



output 20 15 10 5