Doubly linked list

←: Global Structure :→

```
struct node
{
    struct node * next;
    int data;
    struct node *prev;
};
```

```
struct node *create_ll (struct node *start)
{
  struct node * new_node, *ptr;
  int num;
  printf ("\n Enter -1 to end");
  printf("\n Enter the data: ");
  scanf ("%d", &num);
  while (num!=-1)
    {
      if (start == NULL)
        {
          new_node = (struct node *) malloc (sizeof (struct node));
          new_node -> prev = NULL;
          new_node -> data = num;
          new_node -> next = NULL;
           start = new_node;
        }
      else
        {
          ptr = start;
          new_node = (struct node*) malloc (sizeof(struct node));
          new_node -> data = num;
          while (ptr -> next != NULL)
                { ptr = ptr -> next;
                }
           ptr -> next = new_node;
            new_node -> prev = ptr;
            new_node -> next = NULL;
        }
      printf ("\n Enter the data:");
       scanf ("%d", &num);
    }
    return start;
}
```

```
struct node * display (struct node *start)
  {
   struct node *ptr;
    ptr = start;
   while(ptr!=NULL)
     {
       printf("\t%d", ptr->data);
         ptr = ptr->next;
      }
      return start;
  }

Struct node *insert_beg (struct node *start)
    {
      struct node * new_node;
      int num;
      printf ("\n Enter the data :");
      scanf ("%d", &num);
      new_node = (struct node *) malloc (size of (struct node));
      new_node -> data = num;
      start -> prev = new_node;
      new_node -> next = start;
      new_node -> prev = NULL;
       start = new_node;
       return start;
    }
```

```
struct node * insert_end (struct node * start)
{
    struct node *ptr, *new_node;
    int num;
    printf ("\n Enter the data:");
    scanf ("%d", &num);
    new_node = (struct node *) malloc(sizeof (struct node));
    new_node -> data = num;
    ptr = start;
    while (ptr->next != NULL)
        {
          ptr = ptr -> next;
        }
      ptr -> next = new_node;
      new_node -> prev = ptr;
      new_node -> next = NULL;
       return (start;
}

struct node * insert_before ( struct node * start)
    {
      struct node * new_node, *ptr;
       int num, val;
      printf("\n Enter the data ");
      scanf ("%d", &num);
      printf ("\n Enter the value before which the data has to be
                  inserted");
      scanf ("%d", &val);
       new_node = (struct node*) malloc (sizeof (struct node));
       new_node -> data = num;
       ptr = start;
       while ( ptr->data != val)
             {
               ptr = ptr -> next;
             }
        new_node -> next = ptr;
         new_node -> prev = ptr -> prev;
         ptr -> prev -> next = new_node;
         ptr -> prev = new_node;
      } return start;
```

```
struct node *insert_after( struct node * start)
{
  struct node  *new_node , *ptr;
   int num, val;
  printf ("\n Enter the data :");
  scanf ("%d", &num);
  printf ("\n Enter the value after which the data has to be inserted :");
  &canf ("%d", &val);
   new_node = (struct node*) malloc (sizeof (struct node));
    new_node -> data = num;
     ptr = start;
    while (ptr-> data != val)
        {
          ptr = ptr -> next;
        }
       new_node -> prev = ptr;
        new_node -> next = ptr->next;
         ptr-> next -> prev = new_node;
           ptr-> next = new_node;
            return start;
        }

struct node *delete_beg (struct node *start)
     {
        struct node *ptr;
          ptr = start;
         start = start -> next;
         start -> prev = NULL;
         free(ptr);
         return start;
       }
```

```
struct node * delete_end ( struct node * start)
    {
     struct node *ptr;
      ptr = start;
     while (ptr->next != NULL)
          {
            ptr = ptr -> next;
          }
        ptr -> prev -> next = NULL;
        free (ptr);
        return start;
     }

struct node * delete_after (struct node * start)
      {
        struct node *ptr, *temp;
        int val;
        printf ("\n Enter the value after which the node has to deleted : ");
        scanf ("%d", &val);
         ptr = start;
        while (ptr->data != val)
                  {
                    ptr = ptr->next;
                  }
           temp = ptr->next;
           ptr->next = temp->next ;
           temp->next->prev = ptr;
            free (temp);
             return start;
          }
```

```
struct node * delete_before (struct node *start)
{
 struct node *ptr, *temp;
 int val;
 printf ("\n Enter the value before which the node has to be deleted:");
 scanf ("%d", &val);
  ptr = start;
  while (ptr->data != val)
        {
         ptr = ptr->next;
        }
       temp = ptr->~~next~~ prev;
       ptr->next = temp->next;
        temp->next->prev = ptr;
        free (temp);
        return start;
}
```

```
if (temp == start)
 {
  start = delete_beg (start);
 }
 else
 {
   ptr->prev = temp->prev;
   temp->prev->next = ptr;
 }
 free (temp);
 return start;
}
```

```
Struct node * delete_list (struct node *start)
{
   while (start != NULL)
          start = delete_beg (start);
    return start;
}
```