1)//linkedlist-sort,reverse,concat

```
#include <stdio.h>
#include <stdlib.h>
struct node
{
   int data;
   struct node *next;
}*first=NULL,*second=NULL;
void display(struct node *s)
{
   while(s!=NULL)
   {
      printf("%d\t",s->data);
      s=s->next;
   }
}
void create1(int a[],int n)
{
   struct node *last,*t;
   first=(struct node *) malloc (sizeof(struct node));
   first->data=a[0];
   first->next=NULL;
   last=first;
   for(int i=1;i<n;i++)
   {
      t=(struct node *) malloc (sizeof(struct node));
      t->data=a[i];
      t->next=NULL;
      last->next=t;
      last=t;
   }

}
void create2(int a[],int n)
{
   struct node *last,*t;
   second=(struct node *) malloc (sizeof(struct node));
   second->data=a[0];
   second->next=NULL;
   last=second;
   for(int i=1;i<n;i++)
   {
      t=(struct node *) malloc (sizeof(struct node));
      t->data=a[i];
      t->next=NULL;
      last->next=t;
      last=t;
   }
```

```
}
void reverse(struct node *p)
{
   struct node *q,*r;
   p=first;
   q=NULL;
   r=NULL;
   while(p!=NULL)
   {
      r=q;
      q=p;
      p=p->next;
      q->next=r;

   }
   first=q;
}
struct node* concatat(struct node *p,struct node *q)
{
   struct node *r;
   if(p==NULL)
   {
      p=q;
      return p;
   }
   if(q==NULL)
   {
      return q;
   }
   r=p;
   while(r->next!=NULL)
      r=r->next;
   r->next=q;
   return p;

}
void sort(struct node *p)
{
   struct node *i,*j;
   int temp;
   for(i=p;i->next!=NULL;i=i->next)
   {
      for(j=i->next;j!=NULL;j=j->next)
      {
         if(i->data>j->data)
         {
            temp=i->data;
```

```
            i->data=j->data;
            j->data=temp;
          }
       }
    }
}
void main()
{
   int a[10],b[10],n1,n2;
   printf("enter n:");
   scanf("%d",&n1);
   printf("enter the values");
   for(int i=0;i<n1;i++)
   {
      scanf("%d",&a[i]);
   }
   struct node *s;
   s=(struct node *) malloc (sizeof(struct node));
   create1(a,n1);
   sort(first);
   printf("sorted list");
   display(first);
   reverse(first);
   printf("\nreversed list");
   display(first);
   printf("\nenter n:");
   scanf("%d",&n2);
   printf("enter the values");
   for(int i=0;i<n2;i++)
   {
      scanf("%d",&b[i]);
   }
   create2(b,n2);
   display(second);
   s=concatat(first,second);
   display(s);
}
```

Output

```
enter n:5
enter the values9
5
7
3
1
sorted list1    3       5       7       9
reversed list9  7       5       3       1
enter n:4
enter the values2
9
4
6
2       9       4       6
concat 9        7       5       3       1       2       9       4       6
```

2)//stack implement using linked list

```
#include <stdio.h>
#include <stdlib.h>
struct node
{
   int data;
   struct node *next;
} *top = NULL;
int empty()
{
   if (top == NULL)
      return 1;
   return 0;
}
int full()
{
   struct node *t;
   t = (struct node*)malloc(sizeof(struct node));
   if (t == NULL)
      return 1;
   return 0;
}
void push(int x)
{
   struct node *t;
   t = (struct node*)malloc(sizeof(struct node));

   if (full())
   {
      printf("overflow");
   }
   else
   {
      t->data = x;
      t->next = top;
      top = t;
   }
}
int pop()
{
   struct node *t;
   // t = (*struct node)malloc(sizeof(struct node));
   int x = -1;
   if(empty())
   {
      printf("stack underflow");
      return x;
   }
   else
   {
```

```
        t = top;
        top = top->next;
        x = t->data;
        free(t);
        return x;
    }
}
void display()
{
    struct node *t;
    // t = (*struct node)malloc(sizeof(struct node));
    t=top;
    while(t!=NULL)
    {
        printf("%d\t",t->data);
        t=t->next;
    }
    printf("\n");

}
void main()
{
    int c, no, x;
    while (1)
    {
        printf("enter 1 for insert 2 for delete 3 for display 4 for exit\n");
        printf("enter the choice:");
        scanf("%d", &c);
        switch (c)
        {
        case 1:
            printf("enter the no:");
            scanf("%d", &no);
            push(no);

            break;
        case 2:
            x = pop();
            if (x != -1)
            {
                printf("%d is popped\n", x);
            }
            break;
        case 3:
            display();
            break;
        case 4:
            exit(0);

        default:
```

```
            printf("invalid\n");
            break;
        }
    }
}
```

Output

```
enter 1 for insert 2 for delete 3 for display 4 for exit
enter the choice:1
enter the no:10
enter 1 for insert 2 for delete 3 for display 4 for exit
enter the choice:1
enter the no:20
enter 1 for insert 2 for delete 3 for display 4 for exit
enter the choice:1
enter the no:30
enter 1 for insert 2 for delete 3 for display 4 for exit
enter the choice:3
30      20      10
enter 1 for insert 2 for delete 3 for display 4 for exit
enter the choice:2
30 is popped
enter 1 for insert 2 for delete 3 for display 4 for exit
enter the choice:2
20 is popped
enter 1 for insert 2 for delete 3 for display 4 for exit
enter the choice:4
```

3)//queque implement using linked list

```
#include <stdio.h>
#include <stdlib.h>
struct node
{
    int data;
    struct node *next;
} *front = NULL,*rear=NULL;
void enqueue(int x)
{
    struct node *t;
    t=(struct node*) malloc (sizeof(struct node));
    if(t==NULL)
    {
        printf("overflow");
    }
    else{
        t->data=x;
        t->next=NULL;
        if(front==NULL)
        {
            front=rear=t;
        }
        else
```

```
        {
            rear->next=t;
            rear=t;
        }
    }
}
int dequeue()
{
    struct node *t;
    int x=-1;
    if(front==NULL)
    {
        printf("underflow");
        return x;
    }
    else{
        x=front->data;
        t=front;
        front=front->next;

        free(t);
        return x;

    }


}
void display()
{
    struct node *t;
    // t = (*struct node)malloc(sizeof(struct node));
    t=front;
    while(t)
    {
        printf("%d\t",t->data);
        t=t->next;
    }
    printf("\n");

}
void main()
{
    int c, no, x;
    while (1)
    {
        printf("enter 1 for insert 2 for delete 3 for display 4 for exit\n");
        printf("enter the choice:");
        scanf("%d", &c);
        switch (c)
        {
```

```
        case 1:
            printf("enter the no:");
            scanf("%d", &no);
            enqueue(no);

            break;
        case 2:
            x = dequeue();
            if (x != -1)
            {
                printf("%d is popped\n", x);
            }
            break;
        case 3:
            display();
            break;
        case 4:
            exit(0);

        default:
            printf("invalid\n");
            break;
        }
    }
}
```

Output

```
enter 1 for insert 2 for delete 3 for display 4 for exit
enter the choice:1
enter the no:10
enter 1 for insert 2 for delete 3 for display 4 for exit
enter the choice:1
enter the no:20
enter 1 for insert 2 for delete 3 for display 4 for exit
enter the choice:1
enter the no:30
enter 1 for insert 2 for delete 3 for display 4 for exit
enter the choice:3
10      20      30
enter 1 for insert 2 for delete 3 for display 4 for exit
enter the choice:2
10 is popped
enter 1 for insert 2 for delete 3 for display 4 for exit
enter the choice:2
20 is popped
enter 1 for insert 2 for delete 3 for display 4 for exit
enter the choice:4
```