DATA STRUCTURES

LAB REPORT

NAME: Adithi Girimaji

USN: 1BM19CS005

SUBJECT: Data Structures

ACADEMIC YEAR: 2

1.

```
#include
<stdio.h>
            #define size 3
            int top=-1;
            void push(int [], int);
            int pop(int[]);
            void display(int []);
            int main()
            {
                   int stack[size],choice,element,ch;
                do
                {
                printf("Enter your choice\n");
                printf("1. Push\n2. Pop\n3. Display\n");
                scanf("%d",&choice);
                switch(choice)
                {
                    case 1: printf("Enter the element to be pushed \n");
                            scanf("%d",&element);
                            push(stack,element);
                            break;
                    case 2: element=pop(stack);
                            if(element==-1)
                                printf("Stack Underflow\n");
                            else
                               printf("Poped element is %d \n",element);
                            break;
                    case 3: display(stack);
                            break;
                    default: printf("Wrong choice\n");
                }
                printf("Do you want to continue?:Press 0 to stop, else press any other
            number\n");
                scanf("%d",&ch);
                } while(ch!=0);
                   return 0;
            }

            void push(int stack[], int ele)
            {
                if (top==size-1)
                {
                    printf("Stack overflow. This element cannot be added to stack.\n");
                }
                else
```

```
    {
        top++;
        stack[top]=ele;
    }
}

int pop(int stack[])
{
    int popele;
    if(top==-1)

       return -1;

    else
    {
        popele=stack[top];
        top--;
        return popele;
    }


}

void display(int stack[])
{
       int i;
    printf("The stack elemements are\n");
    for(i=top;i>=0;i--)
    {

        printf("%d\t",stack[i]);
    }
    printf("\n");
}
```

```
Enter your choice
1. Push
2. Pop
3. Display
1
Enter the element to be pushed
1
Do you want to continue?:Press 0 to stop, else press any other number
1
Enter your choice
1. Push
2. Pop
3. Display
1
Enter the element to be pushed
2
Do you want to continue?:Press 0 to stop, else press any other number
1
Enter your choice
1. Push
2. Pop
3. Display
1
Enter the element to be pushed
3
Do you want to continue?:Press 0 to stop, else press any other number
1
Enter your choice
1. Push
2. Pop
3. Display
1
Enter the element to be pushed
4
Stack overflow. This element cannot be added to stack.
Do you want to continue?:Press 0 to stop, else press any other number
1
Enter your choice
1. Push
2. Pop
3. Display
3
The stack elemements are
3       2       1
Do you want to continue?:Press 0 to stop, else press any other number
1
Enter your choice
1. Push
2. Pop
3. Display
2
Poped element is 3
Do you want to continue?:Press 0 to stop, else press any other number
```

```
Enter your choice
1. Push
2. Pop
3. Display
2
Poped element is 3
Do you want to continue?:Press 0 to stop, else press any other number
1
Enter your choice
1. Push
2. Pop
3. Display
2
Poped element is 2
Do you want to continue?:Press 0 to stop, else press any other number
1
Enter your choice
1. Push
2. Pop
3. Display
2
Poped element is 1
Do you want to continue?:Press 0 to stop, else press any other number
1
Enter your choice
1. Push
2. Pop
3. Display
2
Stack Underflow
Do you want to continue?:Press 0 to stop, else press any other number
0


------------------
(program exited with code: 0)

Press any key to continue . . .
```

++

2.

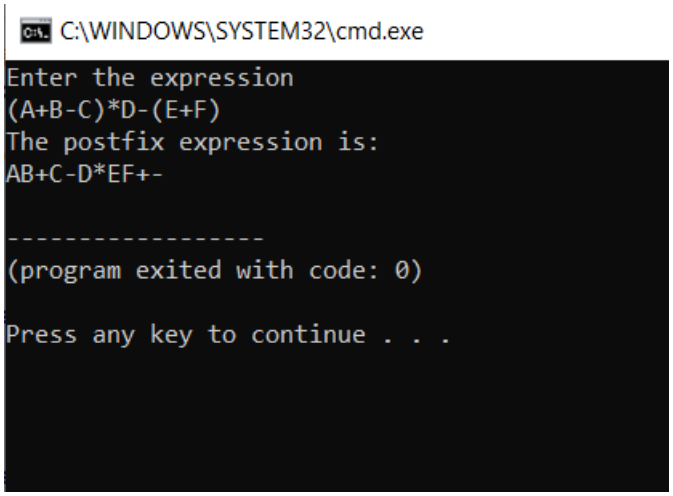
```
#include<stdio.h>
                    #include<string.h>
                    int main()
                    {
                            char stack[20],iexp[50],pexp[50];
                            int i,p=0,s=-
                    1,f,z=1,operands=0,operators=0,obracket=0,cbracket=0;
                            printf("Enter the expression\n");
                            scanf("%s",iexp);
                            for(i=0;i<strlen(iexp);i++)
```

```
	{
		if(iexp[i]=='+'||iexp[i]=='-
'||iexp[i]=='*'||iexp[i]=='/'||iexp[i]=='+'||iexp[i]=='^')
		operators++;

	if((iexp[i]>=65&&iexp[i]<=90)||(iexp[i]>=97&&iexp[i]<=122))
		operands++;
		if(iexp[i]=='(')
		obracket++;
		if(iexp[i]==')')
		cbracket++;
	}
	if(operands!=(operators+1)||obracket!=cbracket)
	z=0;

	for(i=0;i<strlen(iexp);i++)
	{

	if((iexp[i]>=65&&iexp[i]<=90)||(iexp[i]>=97&&iexp[i]<=122))
		{
			pexp[p]=iexp[i];
			p++;
		}
		else if(iexp[i]=='(')
		{
			s++;
			stack[s]=iexp[i];
		}
		else if(iexp[i]==')')
		{
			if(iexp[i-1]=='+'||iexp[i-1]=='-'||iexp[i-
1]=='*'||iexp[i-1]=='/'||iexp[i-1]=='+')
			z=0;
			do
			{
				if(stack[s]=='(')
				{
					s--;
					break;
				}
				pexp[p]=stack[s];
				p++;
				s--;
			}
			while('c'=='c');
		}
```

```
                else if(iexp[i]=='+'||iexp[i]=='-')
                {
                        if(stack[s]=='('||s==-1)
                        {
                                s++;
                                stack[s]=iexp[i];
                        }
                        else
                        {
                                do
                                {
                                        f=0;
                                        if(stack[s]=='(')
                                        {
                                                s++;
                                                stack[s]=iexp[i];
                                                f=1;
                                                break;
                                        }
                                        pexp[p]=stack[s];
                                        p++;
                                    s--;
                                }
                                while(s!=-1);
                                if(f==0)
                                {
                                s++;
                            stack[s]=iexp[i];
                                }
                        }
                }
                else if(iexp[i]=='*'||iexp[i]=='/')
                {
                        if(stack[s]=='('||stack[s]=='+'||stack[s]=='-
'||s==-1)
                        {
                                s++;
                                stack[s]=iexp[i];
                        }
                        else
                        {
                                do
                                {
                                        f=0;
                                        if(stack[s]=='+'||stack[s]=='-
'||stack[s]=='(')
```

```
                                        {
                                                s++;
                                                stack[s]=iexp[i];
                                                f=1;
                                                break;
                                        }
                                        pexp[p]=stack[s];
                                        p++;
                                    s--;
                                }
                                while(s!=-1);
                                if(f==0)
                                {
                                s++;
                            stack[s]=iexp[i];
                            }
                        }
                }
                        else if(iexp[i]=='^')
                        {
                        if(stack[s]=='('||stack[s]=='+'||stack[s]=='-
'||s==-1||stack[s]=='*'||stack[s]=='/')
                        {
                                s++;
                                stack[s]=iexp[i];
                        }
                        else
                        {
                                do
                                {
                                        f=0;
                                        if(stack[s]=='+'||stack[s]=='-
'||stack[s]=='('||stack[s]=='*'||stack[s]=='/')
                                        {
                                                s++;
                                                stack[s]=iexp[i];
                                                f=1;
                                                break;
                                        }
                                        pexp[p]=stack[s];
                                        p++;
                                    s--;
                                }
                                while(s!=-1);
                                if(f==0)
                                {
```

```
                                s++;
                            stack[s]=iexp[i];
                            }
                        }
                }
        }
        if(s!=-1)
        {
                do
                {
                        pexp[p]=stack[s];
                        p++;
                    s--;
                }
                while(s!=-1);
        }
        if(z==0)
        printf("Invalid expression\n");
        else
        {
        printf("The postfix expression is:\n");
        for(i=0;i<p;i++)
        {
                printf("%c",pexp[i]);
        }
    }
        return 0;
}
```



3.

```
#include
<stdio.h>
            #include <stdlib.h>
            #define size 3
            void enque(int [], int, int*);
            void deque(int [],int*,int*);
            void display(int [],int*,int*);
            int main()
            {
                   int queue[size],choice,element,ch, rear=-1,front=0;
                do
                {
                printf("Enter your choice\n");
                printf("1. Insert\n2. Delete\n3. Display\n");
                scanf("%d",&choice);
                switch(choice)
                {
                    case 1: printf("Enter the element to be inserted \n");
                            scanf("%d",&element);
                            enque(queue,element,&rear);
                            break;
                    case 2: deque(queue,&rear,&front);
                            break;
                    case 3: display(queue,&rear,&front);
                            break;
                    default: printf("Wrong choice\n");
                }
                printf("Do you want to continue?:Press 0 to stop, else press any other
            number\n");
                scanf("%d",&ch);
                } while(ch!=0);
                   return 0;
            }

            void enque(int queue[], int ele,int *prear)
            {
                if (*prear==size-1)
                {
                    printf("Queue overflow. This element cannot be added to the
            queue.\n");
                }
                else
                {
                    (*prear)++;
                    queue[*prear]=ele;
```

```
    }
}

void deque(int queue[], int *prear, int *pfront)
{
    if((*prear)==-1 && ((*pfront)==0))
       printf("Queue is empty\n");
    else
    {
        printf("Deleted element is %d \n",queue[*pfront]);
        (*pfront)++;
        if((*pfront)>(*prear))
        {
                      (*pfront)=0;
                      (*prear)=-1;
             }
    }
}

void display(int queue[],int *prear,int *pfront)
{
       int i;
    printf("The queue elemements are\n");
    for(i=(*pfront);i<=(*prear);i++)
    {

        printf("%d\t",queue[i]);
    }
    printf("\n");
}
```

```
Enter your choice
1. Insert
2. Delete
3. Display
1
Enter the element to be inserted
1
Do you want to continue?:Press 0 to stop, else press any other number
1
Enter your choice
1. Insert
2. Delete
3. Display
1
Enter the element to be inserted
2
Do you want to continue?:Press 0 to stop, else press any other number
1
Enter your choice
1. Insert
2. Delete
3. Display
1
Enter the element to be inserted
3
Do you want to continue?:Press 0 to stop, else press any other number
1
Enter your choice
1. Insert
2. Delete
3. Display
1
Enter the element to be inserted
4
Queue overflow. This element cannot be added to stack.
Do you want to continue?:Press 0 to stop, else press any other number
1
Enter your choice
1. Insert
2. Delete
3. Display
3
The queue elemements are
1       2       3
Do you want to continue?:Press 0 to stop, else press any other number
1
Enter your choice
1. Insert
2. Delete
3. Display
2
Deleted element is 1
```

```
Enter your choice
1. Insert
2. Delete
3. Display
2
Deleted element is 1
Do you want to continue?:Press 0 to stop, else press any other number
1
Enter your choice
1. Insert
2. Delete
3. Display
2
Deleted element is 2
Do you want to continue?:Press 0 to stop, else press any other number
1
Enter your choice
1. Insert
2. Delete
3. Display
2
Deleted element is 3
Do you want to continue?:Press 0 to stop, else press any other number
1
Enter your choice
1. Insert
2. Delete
3. Display
2
Queue is empty
Do you want to continue?:Press 0 to stop, else press any other number
0


------------------
(program exited with code: 0)

Press any key to continue . . .
```

++

4.

```
#include
<stdio.h>
            #include <stdlib.h>

            int front=-1;
            int rear=-1;
            int queue[10];
            int MAX;

            void Enque(int);
            void Deque();
            void display();
```

```
int main()
{
        int choice,item;
        printf("Enter the maximum number of elements in the queue\n");
        scanf("%d",&MAX);
    do{
        printf("1. Insert into the queue\n");
        printf("2. Delete from the queue\n");
        printf("3. Display the contents of the queue\n");
        printf("4. Exit\n");
        printf("Enter your choice:\n");
        scanf("%d",&choice);
        switch(choice)
        {
            case 1:  if(((front==0 && rear==MAX-1))||(front==rear+1))
                     {
                                              printf("Queue is full.\n");
                                    }
                                    else
                                    {
                     printf("Enter the element\n");
                     scanf("%d",&item);
                     Enque(item);
                                 }
                     break;
            case 2: Deque();break;
            case 3: display();
                    break;
            case 4: exit(0);
        }
    } while (choice!=4);
       return 0;
}

void Enque(int ele)
{

      rear=(rear+1)%MAX;
      queue[rear]=ele;
      if(front ==-1)
          front=0;
      return;


}
void Deque()
```

```
{
    int item;
    if((front==-1)&&(rear==-1))
    {

        printf("Queue is empty\n");return;
    }
    else
    {
        item=queue[front];

        if(front==rear)
        {
            front=-1;
            rear=-1;
        }
        else
        {
            front=(front+1)%MAX;
        }
    }
    printf("Removed element from the queue is %d\n",item);
    return;
}

void display()
{
    int i;
    if(((front==-1)&&(rear==-1)))
    {

        printf("Queue is empty\n");return;

    }
    else
    {
        printf("Queue contents:\n");
        for(i=front;i!=rear;i=(i+1)%MAX)
           {
            printf("%d\t", queue[i]);
           }
           printf("%d",queue[rear]);
        printf("\n");
        return;
    }
}
```

```
Enter the maximum number of elements in the queue
3
1. Insert into the queue
2. Delete from the queue
3. Display the contents of the queue
4. Exit
Enter your choice:
1
Enter the element
344
1. Insert into the queue
2. Delete from the queue
3. Display the contents of the queue
4. Exit
Enter your choice:
1
Enter the element
656
1. Insert into the queue
2. Delete from the queue
3. Display the contents of the queue
4. Exit
Enter your choice:
1
Enter the element
787
1. Insert into the queue
2. Delete from the queue
3. Display the contents of the queue
4. Exit
Enter your choice:
1
Queue is full.
1. Insert into the queue
2. Delete from the queue
3. Display the contents of the queue
4. Exit
Enter your choice:
3
Queue contents:
344     656     787
1. Insert into the queue
2. Delete from the queue
3. Display the contents of the queue
4. Exit
Enter your choice:
2
Removed element from the queue is 344
1. Insert into the queue
2. Delete from the queue
```

```
2. Delete from the queue
3. Display the contents of the queue
4. Exit
Enter your choice:
1
Enter the element
988
1. Insert into the queue
2. Delete from the queue
3. Display the contents of the queue
4. Exit
Enter your choice:
3
Queue contents:
656     787     988
1. Insert into the queue
2. Delete from the queue
3. Display the contents of the queue
4. Exit
Enter your choice:
2
Removed element from the queue is 656
1. Insert into the queue
2. Delete from the queue
3. Display the contents of the queue
4. Exit
Enter your choice:
2
Removed element from the queue is 787
1. Insert into the queue
2. Delete from the queue
3. Display the contents of the queue
4. Exit
Enter your choice:
2
Removed element from the queue is 988
1. Insert into the queue
2. Delete from the queue
3. Display the contents of the queue
4. Exit
Enter your choice:
2
Queue is empty
1. Insert into the queue
2. Delete from the queue
3. Display the contents of the queue
4. Exit
Enter your choice:
4


------------------
```

5.

```
#include
<stdio.h>
            #include <stdlib.h>
            void create();
            void display();
            void insert_pos1();
            void insert_bef();
            void insert_aft();
            void insert_last();
            void insert_pos();
            struct node
            {
                int id;
                char name[50];
                int sem;
                struct node *next;
            };
            struct node *head=NULL;
            int main(int argc, char **argv)
            {
                int choice,ch;
                do
                {
                printf("1. Create \n2. Display  \n3. Insert at position 1  \n4. Insert
            before an element \n5. Insert after an element\n6. Insert at the end of the
            list\n7. Insert at any position mentioned \n");
                printf("\nEnter your choice : ");
                scanf("%d",&choice);
                switch(choice)
                {
                    case 1: create();break;
                    case 2: display();break;
                    case 3: insert_pos1();break;
                    case 4: insert_bef();break;
                    case 5: insert_aft();break;
                    case 6: insert_last(); break;
                    case 7: insert_pos();break;
                    default: printf("Wrong choice");
                }
                   printf("\nPress 1 if you want to continue else any other number\n");
                scanf("%d",&ch);
                }while(ch==1);
                return 0;
            }
```

```
void create()
{
   struct node *newnode,*temp;
    int ID,s,j;
    char n[50];
    newnode =(struct node *) malloc (sizeof(struct node));
    printf("Enter the student id,name and sem : ");
    scanf("%d",&ID);
    scanf("%s",n);
    scanf("%d",&s);
    newnode->id=ID;
    for(j=0;;j++)
    {
               newnode->name[j]=n[j];
               if(n[j]=='\0')
               break;
       }
    newnode->sem=s;
    if (head==NULL)
     {
      newnode->next=NULL;
      head=newnode;
      printf("Node is created\n");
     }
     else
    {
    temp=head;
       while(temp->next!=NULL)
       {
          temp=temp->next;
       }
      temp->next=newnode;
      newnode->next=NULL;
      printf("Node is created\n");
   }
}

void display()
{
    struct node *ptr=NULL;
    ptr=head;

    if(ptr==NULL)
    {
        printf("Nothing to print\n");
    }
```

```
    else
    {
        while(ptr!=NULL)
       {
        printf("\tID:%d\tName:%s\tSem:%d\n ",ptr->id,ptr->name,ptr->sem);
        ptr=ptr->next;
       }
    }

}
void insert_pos1()
{
    struct node *newnode;
    int ID,s,j;
    char n[50];
    printf("Enter the student id,name and sem : ");
    scanf("%d",&ID);
    scanf("%s",n);
    scanf("%d",&s);
    newnode=(struct node*)malloc(sizeof(struct node));
    newnode->id=ID;
    for(j=0;;j++)
    {
               newnode->name[j]=n[j];
               if(n[j]=='\0')
               break;
       }
    newnode->sem=s;
    newnode->next=head;
    head=newnode;
}

void insert_bef()
{
    struct node *newnode,*temp1,*temp2=NULL;
       temp1=head;
    int ID,s,ele_bef,j;
    char n[50];
    printf("Enter the student ID before which the data has to be inserted:
");
    scanf("%d",&ele_bef);
       printf("Enter the student id,name and sem : ");
    scanf("%d",&ID);
    scanf("%s",n);
    scanf("%d",&s);
    newnode=(struct node*)malloc(sizeof(struct node));
```

```
    newnode->id=ID;
    for(j=0;;j++)
    {
               newnode->name[j]=n[j];
               if(n[j]=='\0')
               break;
       }
    newnode->sem=s;
    if(head->id==ele_bef)
    {
               newnode->next=head;
        head=newnode;
        return;
       }
       while(temp1->next!=NULL)
       {
               if(temp1->next->id==ele_bef)
               {
                       temp2=temp1->next;
                temp1->next=newnode;
                       newnode->next=temp2;
                       return;
               }
               else
                       temp1=temp1->next;
       }
       if(temp2==NULL)
    {
        printf("Element is not found in the list\n");return;
       }
}

void insert_aft()
{
       struct node *newnode,*temp1,*temp2=NULL;
       temp1=head;
    int ID,ele_aft,s,j;
    char n[50];
       printf("Enter the student ID after which the data has to be
inserted: ");
    scanf("%d",&ele_aft);
       printf("Enter the student id,name and sem : ");
    scanf("%d",&ID);
    scanf("%s",n);
    scanf("%d",&s);
    newnode=(struct node*)malloc(sizeof(struct node));
```

```
    newnode->id=ID;
    for(j=0;;j++)
    {
              newnode->name[j]=n[j];
              if(n[j]=='\0')
              break;
       }
    newnode->sem=s;
       while(temp1->next!=NULL)
       {
              if(temp1->id==ele_aft)
              {
                     temp2=temp1->next;
                     temp1->next=newnode;
                     newnode->next=temp2;
                     return;
              }
              else
                     temp1=temp1->next;
       }
       if(temp1->id==ele_aft)
       {
              temp1->next=newnode;
              newnode->next=NULL;
       }return;

       if(temp2==NULL)
    {
        printf("Element is not found in the list\n");return;
       }
}
void insert_last()
{
       struct node *newnode,*ptr;
       ptr=head;
    int ID,s,j;
    char n[50];
    printf("Enter the student id,name and sem : ");
    scanf("%d",&ID);
    scanf("%s",n);
    scanf("%d",&s);
    newnode=(struct node*)malloc(sizeof(struct node));
    newnode->id=ID;
    for(j=0;;j++)
    {
              newnode->name[j]=n[j];
```

```
                if(n[j]=='\0')
                break;
        }
    newnode->sem=s;
        do
        {
            if(ptr->next==NULL)
            {
                        ptr->next=newnode;
                        newnode->next=NULL;
                        return;
                }
                ptr=ptr->next;
        }while(1);
}
void insert_pos()
{
        int pos,c=1,j,ID,s;
        char n[50];
        struct node *temp;
        temp=head;
        if(temp!=NULL&&temp->next!=NULL)
        {
        do
        {
            if(temp->next!=NULL)
            {
                        temp=temp->next;
                        c++;
                }
                if(temp->next==NULL)
                break;
        }while(1);
    }
        printf("Enter the position at which the element has to be
inserted\n");
        scanf("%d",&pos);
        if(pos<1||pos>c+1)
        {
                printf("Element cannot be inserted\n");
                return;
        }
        struct node *newnode;
    newnode =(struct node *) malloc (sizeof(struct node));
    printf("Enter the student id,name and sem : ");
    scanf("%d",&ID);
```

```
    scanf("%s",n);
    scanf("%d",&s);
    newnode->id=ID;
    for(j=0;;j++)
    {
              newnode->name[j]=n[j];
              if(n[j]=='\0')
              break;
       }
    newnode->sem=s;
       if(pos==1)
       {
              newnode->next=head;
        head=newnode;
       }
       else if(pos==c+1)
       {
              temp=head;
              do
       {
           if(temp->next==NULL)
           {
                     temp->next=newnode;
                     newnode->next=NULL;
                     return;
              }
              temp=temp->next;
       }while(1);
    }
       else
       {
              struct node *temp2;
              temp=head;
              for(int i=1;i<pos-1;i++)
              {
                     temp=temp->next;
              }
              temp2=temp->next;
           temp->next=newnode;
              newnode->next=temp2;
       }
}
```

```
1. Create
2. Display
3. Insert at position 1
4. Insert before an element
5. Insert after an element
6. Insert at the end of the list
7. Insert at any position mentioned

Enter your choice : 1
Enter the student id,name and sem : 123
qwe
2
Node is created

Press 1 if you want to continue else any other number
1
1. Create
2. Display
3. Insert at position 1
4. Insert before an element
5. Insert after an element
6. Insert at the end of the list
7. Insert at any position mentioned

Enter your choice : 1
Enter the student id,name and sem : 234
wrt
2
Node is created

Press 1 if you want to continue else any other number
1
1. Create
2. Display
3. Insert at position 1
4. Insert before an element
5. Insert after an element
6. Insert at the end of the list
7. Insert at any position mentioned

Enter your choice : 1
Enter the student id,name and sem : 565
ffgg
2
Node is created

Press 1 if you want to continue else any other number
1
1. Create
2. Display
```

```
Press 1 if you want to continue else any other number
1
1. Create
2. Display
3. Insert at position 1
4. Insert before an element
5. Insert after an element
6. Insert at the end of the list
7. Insert at any position mentioned

Enter your choice : 2
        ID:123  Name:qwe        Sem:2
        ID:234  Name:wrt        Sem:2
        ID:565  Name:ffgg       Sem:2

Press 1 if you want to continue else any other number
1
1. Create
2. Display
3. Insert at position 1
4. Insert before an element
5. Insert after an element
6. Insert at the end of the list
7. Insert at any position mentioned

Enter your choice : 3
Enter the student id,name and sem : 444
dfgf
2

Press 1 if you want to continue else any other number
1
1. Create
2. Display
3. Insert at position 1
4. Insert before an element
5. Insert after an element
6. Insert at the end of the list
7. Insert at any position mentioned

Enter your choice : 2
        ID:444  Name:dfgf       Sem:2
        ID:123  Name:qwe        Sem:2
        ID:234  Name:wrt        Sem:2
        ID:565  Name:ffgg       Sem:2

Press 1 if you want to continue else any other number
1
1. Create
```

```
Press 1 if you want to continue else any other number
1
1. Create
2. Display
3. Insert at position 1
4. Insert before an element
5. Insert after an element
6. Insert at the end of the list
7. Insert at any position mentioned

Enter your choice : 4
Enter the student ID before which the data has to be inserted: 123
Enter the student id,name and sem : 555
fds
2

Press 1 if you want to continue else any other number
1
1. Create
2. Display
3. Insert at position 1
4. Insert before an element
5. Insert after an element
6. Insert at the end of the list
7. Insert at any position mentioned

Enter your choice : 2
        ID:444  Name:dfgf       Sem:2
        ID:555  Name:fds        Sem:2
        ID:123  Name:qwe        Sem:2
        ID:234  Name:wrt        Sem:2
        ID:565  Name:ffgg       Sem:2

Press 1 if you want to continue else any other number
1
1. Create
2. Display
3. Insert at position 1
4. Insert before an element
5. Insert after an element
6. Insert at the end of the list
7. Insert at any position mentioned

Enter your choice : 5
Enter the student ID after which the data has to be inserted: 234
Enter the student id,name and sem : 666
wwe
4

Press 1 if you want to continue else any other number
```

```
Press 1 if you want to continue else any other number
1
1. Create
2. Display
3. Insert at position 1
4. Insert before an element
5. Insert after an element
6. Insert at the end of the list
7. Insert at any position mentioned

Enter your choice : 2
        ID:444  Name:dfgf       Sem:2
        ID:555  Name:fds        Sem:2
        ID:123  Name:qwe        Sem:2
        ID:234  Name:wrt        Sem:2
        ID:666  Name:wwe        Sem:4
        ID:565  Name:ffgg       Sem:2

Press 1 if you want to continue else any other number
1
1. Create
2. Display
3. Insert at position 1
4. Insert before an element
5. Insert after an element
6. Insert at the end of the list
7. Insert at any position mentioned

Enter your choice : 6
Enter the student id,name and sem : 888
wer
2

Press 1 if you want to continue else any other number
1
1. Create
2. Display
3. Insert at position 1
4. Insert before an element
5. Insert after an element
6. Insert at the end of the list
7. Insert at any position mentioned

Enter your choice : 2
        ID:444  Name:dfgf       Sem:2
        ID:555  Name:fds        Sem:2
        ID:123  Name:qwe        Sem:2
        ID:234  Name:wrt        Sem:2
        ID:666  Name:wwe        Sem:4
        ID:565  Name:ffgg       Sem:2
```

```
5. Insert after an element
6. Insert at the end of the list
7. Insert at any position mentioned

Enter your choice : 2
        ID:444  Name:dfgf       Sem:2
        ID:555  Name:fds        Sem:2
        ID:123  Name:qwe        Sem:2
        ID:234  Name:wrt        Sem:2
        ID:666  Name:wwe        Sem:4
        ID:565  Name:ffgg       Sem:2
        ID:888  Name:wer        Sem:2

Press 1 if you want to continue else any other number
1
1. Create
2. Display
3. Insert at position 1
4. Insert before an element
5. Insert after an element
6. Insert at the end of the list
7. Insert at any position mentioned

Enter your choice : 2
        ID:444  Name:dfgf       Sem:2
        ID:555  Name:fds        Sem:2
        ID:123  Name:qwe        Sem:2
        ID:234  Name:wrt        Sem:2
        ID:666  Name:wwe        Sem:4
        ID:565  Name:ffgg       Sem:2
        ID:888  Name:wer        Sem:2

Press 1 if you want to continue else any other number
1
1. Create
2. Display
3. Insert at position 1
4. Insert before an element
5. Insert after an element
6. Insert at the end of the list
7. Insert at any position mentioned

Enter your choice : 7
Enter the position at which the element has to be inserted
2
Enter the student id,name and sem : 999
ere
2

Press 1 if you want to continue else any other number
```

```
Press 1 if you want to continue else any other number
1
1. Create
2. Display
3. Insert at position 1
4. Insert before an element
5. Insert after an element
6. Insert at the end of the list
7. Insert at any position mentioned

Enter your choice : 7
Enter the position at which the element has to be inserted
2
Enter the student id,name and sem : 999
ere
2

Press 1 if you want to continue else any other number
1
1. Create
2. Display
3. Insert at position 1
4. Insert before an element
5. Insert after an element
6. Insert at the end of the list
7. Insert at any position mentioned

Enter your choice : 2
        ID:444  Name:dfgf       Sem:2
        ID:999  Name:ere        Sem:2
        ID:555  Name:fds        Sem:2
        ID:123  Name:qwe        Sem:2
        ID:234  Name:wrt        Sem:2
        ID:666  Name:wwe        Sem:4
        ID:565  Name:ffgg       Sem:2
        ID:888  Name:wer        Sem:2

Press 1 if you want to continue else any other number
0


------------------
(program exited with code: 0)

Press any key to continue . . .
```

6.

```
#include
<stdio.h>
            #include <stdlib.h>
            void create();
            void display();
            void delete_pos1();
            void delete();
            void delete_last();
            struct node
            {
                int id;
                char name[50];
                int sem;
                struct node *next;
            };
            struct node *head=NULL;
            int main(int argc, char **argv)
            {
                int choice,ch;
                do
                {
                printf("1. Create \n2. Display  \n3. Delete the first element\n4. Delete
            an element mentioned\n5.Delete the last element\n");
                printf("\nEnter your choice : ");
                scanf("%d",&choice);
                switch(choice)
                {
                    case 1: create();break;
                    case 2: display();break;
                    case 3: delete_pos1();break;
                    case 4: delete();break;
                    case 5: delete_last(); break;
                    default: printf("Wrong choice");
                }
                   printf("\nPress 1 if you want to continue else any other number\n");
                scanf("%d",&ch);
                }while(ch==1);
                return 0;
            }

            void create()
            {
               struct node *newnode,*temp;
                int ID,s,j;
                char n[50];
```

```
    newnode =(struct node *) malloc (sizeof(struct node));
    printf("Enter the student id,name and sem : ");
    scanf("%d",&ID);
    scanf("%s",n);
    scanf("%d",&s);
    newnode->id=ID;
    for(j=0;;j++)
    {
              newnode->name[j]=n[j];
              if(n[j]=='\0')
              break;
       }
    newnode->sem=s;
    if (head==NULL)
     {
      newnode->next=NULL;
      head=newnode;
      printf("Node is created\n");
     }
     else
    {
    temp=head;
       while(temp->next!=NULL)
       {
          temp=temp->next;
       }
      temp->next=newnode;
      newnode->next=NULL;
      printf("Node is created\n");
   }
}

void display()
{
    struct node *ptr=NULL;
    ptr=head;

    if(ptr==NULL)
    {
        printf("Nothing to print\n");
    }
    else
    {
        while(ptr!=NULL)
       {
        printf("\tID:%d\tName:%s\tSem:%d\n ",ptr->id,ptr->name,ptr->sem);
```

```
        ptr=ptr->next;
       }
    }

}
void delete_pos1()
{
       if (head == NULL)
     {
         printf("Empty List. Can't delete\n");return;
     }
     else
     {
               head=head->next;
        }
}

void delete()
{
       int ele;
    printf("Enter the student ID which has to be deleted\n");
    scanf("%d",&ele);
    struct node *temp,*del=NULL;

     if (head == NULL)
     {
         printf("Empty List. Can't delete\n");return;
     }
     temp=head;
     if(temp->id==ele)
     {
              head=head->next;
              return;
        }
    while (temp->next!=NULL)
    {
        if(temp->next->id==ele)
        {
            del=temp->next;
           if(del->next==NULL)
           temp->next=NULL;
           else
            temp->next=del->next;
        }

        else
```

```
            temp=temp->next;
    }
    if(del==NULL)
    {
        printf("Element not found in the list\n");return;
  }
}

void delete_last()
{
       struct node *ptr;
       ptr=head;
       if (head == NULL)
    {
        printf("Empty List. Can't delete\n");return;
    }
    else if(ptr->next==NULL)
    {
              head=NULL;
       }
    else
    {
       do
       {
           if(ptr->next->next==NULL)
           {
                     ptr->next=NULL;
                     return;
              }
              ptr=ptr->next;
       }while(1);
    }
}
```

```
1. Create
2. Display
3. Delete the first element
4. Delete an element mentioned
5.Delete the last element

Enter your choice : 1
Enter the student id,name and sem : 56789
wer
2
Node is created

Press 1 if you want to continue else any other number
1
1. Create
2. Display
3. Delete the first element
4. Delete an element mentioned
5.Delete the last element

Enter your choice : 1
Enter the student id,name and sem : 98989
cvb
2
Node is created

Press 1 if you want to continue else any other number
1
1. Create
2. Display
3. Delete the first element
4. Delete an element mentioned
5.Delete the last element

Enter your choice : 1
Enter the student id,name and sem : 34532
fgh
2
Node is created

Press 1 if you want to continue else any other number
1
1. Create
2. Display
3. Delete the first element
4. Delete an element mentioned
5.Delete the last element

Enter your choice : 1
Enter the student id,name and sem : 22222
```

```
Enter your choice : 1
Enter the student id,name and sem : 22222
fgh
2
Node is created

Press 1 if you want to continue else any other number
1
1. Create
2. Display
3. Delete the first element
4. Delete an element mentioned
5.Delete the last element

Enter your choice : 1
Enter the student id,name and sem : 77777
asd
2
Node is created

Press 1 if you want to continue else any other number
1
1. Create
2. Display
3. Delete the first element
4. Delete an element mentioned
5.Delete the last element

Enter your choice : 1
Enter the student id,name and sem : 67533
jkl
2
Node is created

Press 1 if you want to continue else any other number
1
1. Create
2. Display
3. Delete the first element
4. Delete an element mentioned
5.Delete the last element

Enter your choice : 2
        ID:56789        Name:wer        Sem:2
        ID:98989        Name:cvb        Sem:2
        ID:34532        Name:fgh        Sem:2
        ID:22222        Name:fgh        Sem:2
        ID:77777        Name:asd        Sem:2
        ID:67533        Name:jkl        Sem:2
```

```
        ID:22222        Name:fgh        Sem:2
        ID:77777        Name:asd        Sem:2
        ID:67533        Name:jkl        Sem:2

Press 1 if you want to continue else any other number
1
1. Create
2. Display
3. Delete the first element
4. Delete an element mentioned
5.Delete the last element

Enter your choice : 3

Press 1 if you want to continue else any other number
1
1. Create
2. Display
3. Delete the first element
4. Delete an element mentioned
5.Delete the last element

Enter your choice : 2
        ID:98989        Name:cvb        Sem:2
        ID:34532        Name:fgh        Sem:2
        ID:22222        Name:fgh        Sem:2
        ID:77777        Name:asd        Sem:2
        ID:67533        Name:jkl        Sem:2

Press 1 if you want to continue else any other number
1
1. Create
2. Display
3. Delete the first element
4. Delete an element mentioned
5.Delete the last element

Enter your choice : 4
Enter the student ID which has to be deleted
77777

Press 1 if you want to continue else any other number
1
1. Create
2. Display
3. Delete the first element
4. Delete an element mentioned
5.Delete the last element

Enter your choice : 2
```

```
4. Delete an element mentioned
5.Delete the last element

Enter your choice : 2
        ID:98989        Name:cvb        Sem:2
        ID:34532        Name:fgh        Sem:2
        ID:22222        Name:fgh        Sem:2
        ID:67533        Name:jkl        Sem:2

Press 1 if you want to continue else any other number
1
1. Create
2. Display
3. Delete the first element
4. Delete an element mentioned
5.Delete the last element

Enter your choice : 5

Press 1 if you want to continue else any other number
1
1. Create
2. Display
3. Delete the first element
4. Delete an element mentioned
5.Delete the last element

Enter your choice : 2
        ID:98989        Name:cvb        Sem:2
        ID:34532        Name:fgh        Sem:2
        ID:22222        Name:fgh        Sem:2

Press 1 if you want to continue else any other number
0


------------------
(program exited with code: 0)

Press any key to continue . . .
```

7.

```
#include
<stdio.h>
            #include <stdlib.h>
            void create();
            void display();
            void sort();
            void reverse();
            void concatenate();
            struct node
            {
                int data;
                struct node *next;
            };
            struct node *head=NULL;
            int main(int argc, char **argv)
            {
                int choice,ch;
                do
                {
                printf("1. Create \n2. Display  \n3. Sort in ascending order\n4. Reverse
            \n5. Concatenate 2 linked lists\n");
                printf("\nEnter your choice : ");
                scanf("%d",&choice);
                switch(choice)
                {
                    case 1: create();break;
                    case 2: display();break;
                    case 3: sort();break;
                    case 4: reverse();break;
                    case 5: concatenate(); break;
                    default: printf("Wrong choice");
                }
                   printf("\nPress 1 if you want to continue else any other number\n");
                scanf("%d",&ch);
                }while(ch==1);
                return 0;
            }

            void create()
            {
               struct node *newnode,*temp;
                int item;
                newnode =(struct node *) malloc (sizeof(struct node));
                printf("Enter the data : ");
                scanf("%d",&item);
```

```
    newnode->data=item;
    if (head==NULL)
     {
      newnode->next=NULL;
      head=newnode;
      printf("Node is created\n");
     }
     else
    {
    temp=head;
       while(temp->next!=NULL)
       {
          temp=temp->next;
       }
      temp->next=newnode;
      newnode->next=NULL;
      printf("Node is created\n");
   }
}

void display()
{
    struct node *ptr;
    ptr=head;

    if(ptr==NULL)
    {
        printf("Nothing to print\n");
    }
    else
    {
        while(ptr!=NULL)
       {
        printf("%d ",ptr->data);
        ptr=ptr->next;
       }
    }

}
void reverse()
{
    struct node *prev=NULL,*current=head, *next=NULL;
    while(current!=NULL)
    {
        next=current->next;
        current->next=prev;
```

```
        prev=current;
        current=next;
    }
    head=prev;
}
void sort()
{
       int swapped,temp;
       struct node *ptr1;
       struct node *lptr = NULL;
       if(head==NULL)
       {
              printf("List is empty\n");
              return;
       }
       if(head->next==NULL)
       {
              printf("Sorted list:\n");
              printf("%d\n",head->data);
              return;
       }
    do
    {
        swapped = 0;
        ptr1 = head;

        while (ptr1->next!=lptr)
        {
            if (ptr1->data > ptr1->next->data)
            {
                temp = ptr1->data;
                ptr1->data = ptr1->next->data;
                ptr1->next->data = temp;

                swapped = 1;
            }
            ptr1 = ptr1->next;
        }
        lptr = ptr1;
    }
    while (swapped);
}
void concatenate()
{
struct node *head1=NULL;
struct node *head2=NULL;
```

```
    struct node *newnode1,*temp1;
    struct node *newnode2,*temp2;
 int item,f=1;
 printf("---First list---\n");
 do
 {
 printf("Enter the data : ");
 scanf("%d",&item);
 newnode1 =(struct node *) malloc (sizeof(struct node));
 newnode1->data=item;
 if (head1==NULL)
  {
   newnode1->next=NULL;
   head1=newnode1;
   printf("Node is created\n");
  }
  else
 {
 temp1=head1;
    while(temp1->next!=NULL)
    {
       temp1=temp1->next;
    }
   temp1->next=newnode1;
   newnode1->next=NULL;
   printf("Node is created\n");
}
    printf("Press 1 if you want to continue else any other number\n");
 scanf("%d",&f);
 }
    while(f==1);
    printf("---Second list---\n");
 do
 {
 printf("Enter the data : ");
 scanf("%d",&item);
 newnode2 =(struct node *) malloc (sizeof(struct node));
 newnode2->data=item;
 if (head2==NULL)
  {
   newnode2->next=NULL;
   head2=newnode2;
   printf("Node is created\n");
  }
  else
 {
```

```
    temp2=head2;
       while(temp2->next!=NULL)
       {
          temp2=temp2->next;
       }
      temp2->next=newnode2;
      newnode2->next=NULL;
      printf("Node is created\n");
  }
      printf("Press 1 if you want to continue else any other number\n");
   scanf("%d",&f);
     }
       while(f==1);


       temp1=head1;
       while(temp1->next!=NULL)
       {
              temp1=temp1->next;
       }
       temp1->next=head2;
       printf("Concatenated list:\n");

       struct node *ptr1;
    ptr1=head1;

    if(ptr1==NULL)
    {
        printf("Nothing to print\n");
    }
    else
    {
        while(ptr1!=NULL)
       {
        printf("%d\t",ptr1->data);
        ptr1=ptr1->next;
       }
    }
}
```

```
1. Create
2. Display
3. Sort in ascending order
4. Reverse
5. Concatenate 2 linked lists

Enter your choice : 1
Enter the data : 2
Node is created

Press 1 if you want to continue else any other number
1
1. Create
2. Display
3. Sort in ascending order
4. Reverse
5. Concatenate 2 linked lists

Enter your choice : 1
Enter the data : 4
Node is created

Press 1 if you want to continue else any other number
1
1. Create
2. Display
3. Sort in ascending order
4. Reverse
5. Concatenate 2 linked lists

Enter your choice : 1
Enter the data : 3
Node is created

Press 1 if you want to continue else any other number
1
1. Create
2. Display
3. Sort in ascending order
4. Reverse
5. Concatenate 2 linked lists

Enter your choice : 2
2 4 3
Press 1 if you want to continue else any other number
1
1. Create
2. Display
3. Sort in ascending order
4. Reverse
```

```
3. Sort in ascending order
4. Reverse
5. Concatenate 2 linked lists

Enter your choice : 3

Press 1 if you want to continue else any other number
1
1. Create
2. Display
3. Sort in ascending order
4. Reverse
5. Concatenate 2 linked lists

Enter your choice : 2
2 3 4
Press 1 if you want to continue else any other number
1
1. Create
2. Display
3. Sort in ascending order
4. Reverse
5. Concatenate 2 linked lists

Enter your choice : 4

Press 1 if you want to continue else any other number
1
1. Create
2. Display
3. Sort in ascending order
4. Reverse
5. Concatenate 2 linked lists

Enter your choice : 2
4 3 2
Press 1 if you want to continue else any other number
1
1. Create
2. Display
3. Sort in ascending order
4. Reverse
5. Concatenate 2 linked lists

Enter your choice : 5
---First list---
Enter the data : 1
Node is created
Press 1 if you want to continue else any other number
1
```

```
4 3 2
Press 1 if you want to continue else any other number
1
1. Create
2. Display
3. Sort in ascending order
4. Reverse
5. Concatenate 2 linked lists

Enter your choice : 5
---First list---
Enter the data : 1
Node is created
Press 1 if you want to continue else any other number
1
Enter the data : 2
Node is created
Press 1 if you want to continue else any other number
1
Enter the data : 3
Node is created
Press 1 if you want to continue else any other number
0
---Second list---
Enter the data : 4
Node is created
Press 1 if you want to continue else any other number
1
Enter the data : 5
Node is created
Press 1 if you want to continue else any other number
0
Concatenated list:
1       2       3       4       5
Press 1 if you want to continue else any other number
0


------------------
(program exited with code: 0)

Press any key to continue . . .
```

8.

```
#include
<stdio.h>
            #include <stdlib.h>
            void push();
            void pop();
            void display_s();
            void insert();
            void delete();
            void display_q();
            struct node
            {
                   int data;
                   struct node *next;
            };
            struct node *head=NULL;
            int main()
            {
                   int choice,ch;
                   printf("---STACK---\n");
                do
                {
                printf("1. Push \n2. Pop  \n3. Display \n");
                printf("\nEnter your choice : ");
                scanf("%d",&choice);
                switch(choice)
                {
                    case 1: push();break;
                    case 2: pop();break;
                    case 3: display_s();break;
                    default: printf("Wrong choice");
                }
                   printf("\nPress 1 if you want to continue else any other number\n");
                scanf("%d",&ch);
                }while(ch==1);

                printf("---QUEUE---\n");
                head=NULL;
                do
                {
                printf("1. Insert \n2. Delete \n3. Display \n");
                printf("\nEnter your choice : ");
                scanf("%d",&choice);
                switch(choice)
                {
                    case 1: insert();break;
```

```
         case 2: delete();break;
         case 3: display_q();break;
         default: printf("Wrong choice");
    }
       printf("\nPress 1 if you want to continue else any other number\n");
    scanf("%d",&ch);
    }while(ch==1);

    return 0;
}
void push()
{
        int item;
        struct node *newnode,*temp;
        newnode=(struct node *)malloc(sizeof(struct node));
        printf("Enter the item to be pushed\n");
        scanf("%d",&item);
        newnode->data=item;
        if(head==NULL)
        {
                newnode->next=NULL;
                head=newnode;
                return;
        }
        temp=head;
        while(temp->next!=NULL)
        {
                temp=temp->next;
        }
        temp->next=newnode;
        newnode->next=NULL;
}
void pop()
{
        if(head==NULL)
        {
                printf("Stack is empty\n");
                return;
        }
        if(head->next==NULL)
        {
                printf("Poped element is %d\n",head->data);
                head=NULL;
                return;
        }
        struct node *temp;
```

```
        temp=head;
        while(temp->next->next!=NULL)
        {
                temp=temp->next;
        }
        printf("Poped element is %d\n",temp->next->data);
        temp->next=NULL;
}
void display_s()
{
    struct node *ptr=NULL;
    ptr=head;

    if(ptr==NULL)
    {
        printf("Nothing to print\n");
    }
    else
    {
        while(ptr!=NULL)
       {
        printf("%d ",ptr->data);
        ptr=ptr->next;
       }
    }

}
void insert()
{
        int item;
        struct node *newnode,*temp;
        newnode=(struct node *)malloc(sizeof(struct node));
        printf("Enter the item to be inserted\n");
        scanf("%d",&item);
        newnode->data=item;
        if(head==NULL)
        {
                newnode->next=NULL;
                head=newnode;
                return;
        }
        temp=head;
        while(temp->next!=NULL)
        {
                temp=temp->next;
        }
```

```
        temp->next=newnode;
        newnode->next=NULL;
}
void delete()
{
        if(head==NULL)
        {
                printf("Queue is empty\n");
                return;
        }
                printf("Deleted element is %d\n",head->data);
                head=head->next;
}
void display_q()
{
    struct node *ptr=NULL;
    ptr=head;

    if(ptr==NULL)
    {
        printf("Nothing to print\n");
    }
    else
    {
        while(ptr!=NULL)
       {
        printf("%d ",ptr->data);
        ptr=ptr->next;
       }
    }

}
```

```
---STACK---
1. Push
2. Pop
3. Display

Enter your choice : 1
Enter the item to be pushed
1

Press 1 if you want to continue else any other number
1
1. Push
2. Pop
3. Display

Enter your choice : 1
Enter the item to be pushed
2

Press 1 if you want to continue else any other number
1
1. Push
2. Pop
3. Display

Enter your choice : 1
Enter the item to be pushed
3

Press 1 if you want to continue else any other number
1
1. Push
2. Pop
3. Display

Enter your choice : 3
1 2 3
Press 1 if you want to continue else any other number
1
1. Push
2. Pop
3. Display

Enter your choice : 2
Poped element is 3

Press 1 if you want to continue else any other number
1
1. Push
2. Pop
```

```
1. Push
2. Pop
3. Display

Enter your choice : 2
Poped element is 2

Press 1 if you want to continue else any other number
1
1. Push
2. Pop
3. Display

Enter your choice : 2
Poped element is 1

Press 1 if you want to continue else any other number
1
1. Push
2. Pop
3. Display

Enter your choice : 2
Stack is empty

Press 1 if you want to continue else any other number
1
1. Push
2. Pop
3. Display

Enter your choice : 3
Nothing to print

Press 1 if you want to continue else any other number
0
---QUEUE---
1. Insert
2. Delete
3. Display

Enter your choice : 1
Enter the item to be inserted
11

Press 1 if you want to continue else any other number
1
1. Insert
2. Delete
3. Display
```

```
3. Display

Enter your choice : 1
Enter the item to be inserted
12

Press 1 if you want to continue else any other number
1
1. Insert
2. Delete
3. Display

Enter your choice : 1
Enter the item to be inserted
13

Press 1 if you want to continue else any other number
1
1. Insert
2. Delete
3. Display

Enter your choice : 3
11 12 13
Press 1 if you want to continue else any other number
1
1. Insert
2. Delete
3. Display

Enter your choice : 2
Deleted element is 11

Press 1 if you want to continue else any other number
1
1. Insert
2. Delete
3. Display

Enter your choice : 2
Deleted element is 12

Press 1 if you want to continue else any other number
1
1. Insert
2. Delete
3. Display

Enter your choice : 2
Deleted element is 13
```

```
Enter your choice : 2
Deleted element is 12

Press 1 if you want to continue else any other number
1
1. Insert
2. Delete
3. Display

Enter your choice : 2
Deleted element is 13

Press 1 if you want to continue else any other number
1
1. Insert
2. Delete
3. Display

Enter your choice : 3
Nothing to print

Press 1 if you want to continue else any other number
1
1. Insert
2. Delete
3. Display

Enter your choice : 2
Queue is empty

Press 1 if you want to continue else any other number
0


------------------
(program exited with code: 0)

Press any key to continue . . .
```

++

9.

```
#include<stdio.h>
                    #include<stdlib.h>
                    struct node
                    {
                           int data;
                           struct node *next;
                           struct node *prev;
                    };
                    struct node *head=NULL;
                    void create()
                    {
```

```
        struct node *new_node,*temp;
        new_node=(struct node*)malloc(sizeof(struct node));
        printf("Enter the item\n");
        scanf("%d",&new_node->data);
        new_node->next=NULL;
        new_node->prev=NULL;
        if(head==NULL)
        {
               head=new_node;
        }
        else
        {
                temp=head;
                while(temp->next!=NULL){
                temp=temp->next;}
                temp->next=new_node;
                new_node->prev=temp;

        }

}
void insert_left()
{
    struct node *newnode,*temp1,*temp2=NULL;
       temp1=head;
    int ele,ele_bef;
    printf("Enter the element before which the data has to be
inserted: ");
    scanf("%d",&ele_bef);
       printf("Enter the element which has to be inserted: ");
    scanf("%d",&ele);
    newnode=(struct node*)malloc(sizeof(struct node));
    newnode->data =ele;
    if(temp1->data==ele_bef)
    {
               temp1->prev=newnode;
               newnode->next=temp1;
               newnode->prev=NULL;
               head=newnode;
       }
       while(temp1->next!=NULL)
       {
               if(temp1->next->data==ele_bef)
               {
                       temp2=temp1->next;
                temp1->next=newnode;
```

```
                        newnode->next=temp2;
                        newnode->prev=temp1;
                        temp2->prev=newnode;
                        return;
                }
                else
                        temp1=temp1->next;
        }
        if(temp2==NULL)
    {
         printf("Element is not found in the list\n");return;
        }
}
void insert_right()
{
        struct node *newnode,*temp1,*temp2=NULL;
        temp1=head;
    int ele,ele_aft;
        printf("Enter the element after which the data has to be
inserted: ");
    scanf("%d",&ele_aft);
        printf("Enter the element which has to be inserted: ");
    scanf("%d",&ele);
    newnode=(struct node*)malloc(sizeof(struct node));
    newnode->data =ele;
        while(temp1->next!=NULL)
        {

                if(temp1->data==ele_aft)
                {
                        temp2=temp1->next;
                        temp1->next=newnode;
                        newnode->prev=temp1;
                        newnode->next=temp2;
                        temp2->prev=newnode;
                        return;
                }
                else
                        temp1=temp1->next;
        }
        if(temp1->data==ele_aft&&temp1->next==NULL)
                {
                        temp1->next=newnode;
                        newnode->prev=temp1;
                        newnode->next=NULL;
                         return;
```

```
            }
        if(temp2==NULL)
    {
        printf("Element is not found in the list\n");return;
        }
}
void del()
{
        int ele;

    struct node *temp,*del=NULL;

     if (head == NULL)
     {
         printf("Empty List. Can't delete\n");return;
     }
     printf("Enter the element to be deleted\n");
    scanf("%d",&ele);
     temp=head;

     if(temp->data==ele&&temp->next==NULL)
        {
                head=NULL;
                return;
        }

     if(temp->data==ele)
     {
               head=head->next;
               head->prev=NULL;
               return;
        }

    while (temp->next!=NULL)
    {
        if(temp->next->data==ele)
        {
            del=temp->next;
           if(del->next==NULL)
           temp->next=NULL;
           else{
            temp->next=del->next;
            del->next->prev=temp;
                  }
        }
```

```
        else
           temp=temp->next;
    }
    if(del==NULL)
    {
        printf("Element not found in the list\n");return;
  }
}

void display()
{
       if(head==NULL){
       printf("Nothing to display\n");
       return;
    }
        struct node *temp;
        temp=head;
        while(temp!=NULL)
        {
               printf("%d\t",temp->data);
               temp=temp->next;
        }
        printf("\n");
}


int main()
{
       int choice;

        while(1)
        {
                  printf("\n");
                     printf(" 1. Create \n");
                     printf(" 2. Insert to the left of an
element\n");
                     printf(" 3. Insert to the right of an
element\n");
                     printf(" 4. Delete \n");
                     printf(" 5. Display\n");
                     printf(" 6. Exit\n");
                     printf("Enter your choice\n");
                     scanf("%d",&choice);
                     switch(choice)
                     {
                              case 1:create();break;
```

```
                                        case 2: insert_left(); break;
                                        case 3: insert_right(); break;
                                        case 4: del(); break;
                                        case 5: display(); break;
                                        case 6: exit(0);break;
                                        default:printf("Wrong choice\n");
                                    }
                                }
                            }
```

```
 1. Create
 2. Insert to the left of an element
 3. Insert to the right of an element
 4. Delete
 5. Display
 6. Exit
Enter your choice
1
Enter the item
1

 1. Create
 2. Insert to the left of an element
 3. Insert to the right of an element
 4. Delete
 5. Display
 6. Exit
Enter your choice
1
Enter the item
2

 1. Create
 2. Insert to the left of an element
 3. Insert to the right of an element
 4. Delete
 5. Display
 6. Exit
Enter your choice
1
Enter the item
3

 1. Create
 2. Insert to the left of an element
 3. Insert to the right of an element
 4. Delete
 5. Display
 6. Exit
Enter your choice
5
1       2       3

 1. Create
 2. Insert to the left of an element
 3. Insert to the right of an element
 4. Delete
 5. Display
 6. Exit
Enter your choice
```

```
Enter your choice
2
Enter the element before which the data has to be inserted: 2
Enter the element which has to be inserted: 22

 1. Create
 2. Insert to the left of an element
 3. Insert to the right of an element
 4. Delete
 5. Display
 6. Exit
Enter your choice
5
1       22      2       3

 1. Create
 2. Insert to the left of an element
 3. Insert to the right of an element
 4. Delete
 5. Display
 6. Exit
Enter your choice
3
Enter the element after which the data has to be inserted: 2
Enter the element which has to be inserted: 33

 1. Create
 2. Insert to the left of an element
 3. Insert to the right of an element
 4. Delete
 5. Display
 6. Exit
Enter your choice
5
1       22      2       33      3

 1. Create
 2. Insert to the left of an element
 3. Insert to the right of an element
 4. Delete
 5. Display
 6. Exit
Enter your choice
4
Enter the element to be deleted
33

 1. Create
 2. Insert to the left of an element
 3. Insert to the right of an element
```

```
 3. Insert to the right of an element
 4. Delete
 5. Display
 6. Exit
Enter your choice
5
1       22      2       3

 1. Create
 2. Insert to the left of an element
 3. Insert to the right of an element
 4. Delete
 5. Display
 6. Exit
Enter your choice
4
Enter the element to be deleted
1

 1. Create
 2. Insert to the left of an element
 3. Insert to the right of an element
 4. Delete
 5. Display
 6. Exit
Enter your choice
4
Enter the element to be deleted
22

 1. Create
 2. Insert to the left of an element
 3. Insert to the right of an element
 4. Delete
 5. Display
 6. Exit
Enter your choice
4
Enter the element to be deleted
33
Element not found in the list

 1. Create
 2. Insert to the left of an element
 3. Insert to the right of an element
 4. Delete
 5. Display
 6. Exit
Enter your choice
4
```

```
 6. Exit
Enter your choice
4
Enter the element to be deleted
3

 1. Create
 2. Insert to the left of an element
 3. Insert to the right of an element
 4. Delete
 5. Display
 6. Exit
Enter your choice
4
Enter the element to be deleted
2

 1. Create
 2. Insert to the left of an element
 3. Insert to the right of an element
 4. Delete
 5. Display
 6. Exit
Enter your choice
4
Empty List. Can't delete

 1. Create
 2. Insert to the left of an element
 3. Insert to the right of an element
 4. Delete
 5. Display
 6. Exit
Enter your choice
5
Nothing to display

 1. Create
 2. Insert to the left of an element
 3. Insert to the right of an element
 4. Delete
 5. Display
 6. Exit
Enter your choice
6


------------------
(program exited with code: 0)
```

10.

```
#include
<stdio.h>
            #include <stdlib.h>
            typedef struct BST
            {
                 int data;
                 struct BST* left;
                 struct BST* right;
            }node;
            node *create();
            void insert(node *,node *);
            void preorder(node *);
            void inorder(node *);
            void postorder(node *);
            int main()
            {
                   char ch;
                   node *root=NULL,*temp;
                   int choice;
                   while(1)
                   {
                          printf("1.Construct\n2.Preorder Traversal\n3.Inorder
            traversal\n4.Postorder traversal\n5.Exit\n");
                          printf("\nEnter your choice\n");
                          scanf("%d",&choice);
                          switch(choice)
                          {
                                 case 1:
                                 do
                                 {
                                        temp=create();
                                        if(root==NULL)
                                        root=temp;
                                        else
                                        insert(root,temp);
                                    printf("\nDo you want to continue(y/n)?\n");
                                    getchar();
                                    scanf("%c",&ch);
                                 }while(ch=='y'||ch=='Y');
                                 break;
                                 case 2:
                                 printf("\nPreorder Traversal: ");
                                 preorder(root);
                                 break;
                                 case 3:
```

```
                        printf("\nInorder Traversal: ");
                        inorder(root);
                        break;
                        case 4:
                        printf("\nPostorder Traversal: ");
                        postorder(root);
                        break;
                        case 5:
                        exit(0);
                }
        }
        return 0;
}
node *create()
{
        node *temp;
        printf("Enter the data:");
        temp=(node*)malloc(sizeof(node));
        scanf("%d",&temp->data);
        temp->left=temp->right=NULL;
        return temp;
}
void insert(node *root,node *temp)
{
        if(temp->data<root->data)
        {
                if(root->left!=NULL)
                insert(root->left,temp);
                else
                root->left=temp;
        }
        if(temp->data>root->data)
        {
                if(root->right!=NULL)
                insert(root->right,temp);
                else
                root->right=temp;
        }
}
void preorder(node *root)
{
        if(root!=NULL)
        {
                printf("%d\t",root->data);
                preorder(root->left);
                preorder(root->right);
```

```
        }
    }
    void inorder(node *root)
    {
        if(root!=NULL)
        {
            inorder(root->left);
            printf("%d\t",root->data);
            inorder(root->right);
        }
    }
    void postorder(node *root)
    {
        if(root!=NULL)
        {
            postorder(root->left);
            postorder(root->right);
            printf("%d\t",root->data);
        }
    }
```

```
1.Construct
2.Preorder Traversal
3.Inorder traversal
4.Postorder traversal
5.Exit

Enter your choice
1
Enter the data:7

Do you want to continue(y/n)?
y
Enter the data:9

Do you want to continue(y/n)?
y
Enter the data:3

Do you want to continue(y/n)?
y
Enter the data:4

Do you want to continue(y/n)?
y
Enter the data:11

Do you want to continue(y/n)?
y
Enter the data:13

Do you want to continue(y/n)?
y
Enter the data:12

Do you want to continue(y/n)?
n

1.Construct
2.Preorder Traversal
3.Inorder traversal
4.Postorder traversal
5.Exit

Enter your choice
2

Preorder Traversal:
7       3       4       9       11      13      12
1.Construct
```

```
5.Exit

Enter your choice
2

Preorder Traversal:
7       3       4       9       11      13      12
1.Construct
2.Preorder Traversal
3.Inorder traversal
4.Postorder traversal
5.Exit

Enter your choice
3

Inorder Traversal:
3       4       7       9       11      12      13
1.Construct
2.Preorder Traversal
3.Inorder traversal
4.Postorder traversal
5.Exit

Enter your choice
4

Postorder Traversal:
4       3       12      13      11      9       7
1.Construct
2.Preorder Traversal
3.Inorder traversal
4.Postorder traversal
5.Exit

Enter your choice
5


------------------
(program exited with code: 0)

Press any key to continue . . .
```