Report on

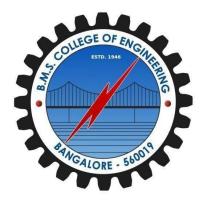
"Data Structures"

Submitted in partial fulfillment of the requirements for the award of the degree of Bachelor of Engineering in Computer Science and Engineering in the course of **Data Structures**(Subject Code)

Submitted by

Medha Madhusudhan (1BM19EC074)

Under the Guidance of **Dr. Kayarvizhy N.**Associate Professor Department of CSE



Department of Computer Science and Engineering
BMS College of Engineering

P.O. Box No.: 1908, Bull Temple Road, Bangalore-560 019 2020-2021

BMSCOLLEGE OF ENGINEERING

P.O. Box No: 1908 Bull Temple Road Bangalore-560019

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING



Report on **Data Structures** (**Subject Code**), "Advanced Algorithm assignment" has been successfully completed by **Medha Madhusudhan** at B.M.S College of Engineering in partial fulfillment of the requirements for the 3rd Semester, degree in Bachelor of Engineering in Computer Science and Engineering under Visvesvaraya Technological University, Belgaum during academic year 2020-2021.

Dr. Kayarvizhy N.

Associate Professor Department of Computer science

Final Marks Awarded

Obtained	Total
	_

CONTENT

SL. No.	CONTENTS	PAGE
		No.
1	Lab Program 1	4
2	Lab Program 2	9
3	Lab Program 3	15
4	Lab Program 4	22
5	Lab Program 5	27
6	Lab Program 6	34
7	Lab Program 7	41
8	Lab Program 8	48
9	Lab Program 9	52
10	Lab Program 10	59

LabProgram 1:

Write a program to simulate the working of stack using an array with the following:

a) Push b) Pop c) Display

The program should print appropriate messages for stack overflow, stack underflow

```
#include <stdio.h>
#include <stdlib.h>
#define MAX 5
int top = -1;
int stack[MAX];
void push(int stack[])
    int element;
    if(top == MAX-1)
        printf("Stack is full!");
    else
            top++;
            printf("Enter element: ");
            scanf("%d", &element);
             stack[top] = element;
        }
}
void pop(int stack[])
    int del;
    if(top == -1)
        printf("Stack is empty!!!\n");
    else
           del = stack[top];
           top--;
           printf("Element deleted is: %d\n",del);
}
void display(int stack[])
```

```
if(top == -1)
        printf("stack is empty!\n");
    else
        for(int i=top;i>=0;i--)
            printf("%d\t",stack[i]);
        printf("\n");
    }
int main(int argc,char** argv)
    int choice;
    while (choice != 4)
        {
            printf("Enter
choice:\n1.Insert\n2.Delete\n3.Display\n4.exit\n");
            scanf("%d", &choice);
            switch(choice)
                case 1:push(stack);
                       break;
                case 2:pop(stack);
                       break;
                case 3:display(stack);
                       break;
                case 4:exit(0);
                default:exit(0);
            }
    return 0;
    }
```

void push (int stack (3)
1
iar element;
-
if (top = = MAX-1)
prints (" stack is feel");
6036
1
top+t',
print; ("Enter element");
Scant ("1.d" element);
Stack [top] = element;
· ·
}
void pop (int stack[])
1
 int de:
ig (top = = -1)
V
cuse
3
· ·
 det = stack [top];

	Classmate Date Page
prints ("Element deleted is: 7.0 \n", del	•
}	
void display (int stack[])	
if (top = = -1) print f ("Stack is empty! 'n");	
else for (int i = top; i >= 0', i) print f (" 1.0 16", stack[i]);	
print f (" \ n");	

```
Element deleted is: 40
Enter choice:
1.Insert
2.Delete
3.Display
4.exit
2
Element deleted is: 30
Enter choice:
1.Insert
2.Delete
3.Display
4.exit
2
Enter choice:
1.Insert
2.Delete
3.Display
4.exit
3
20
10
Enter choice:
```

LabProgram 2:

WAP to convert a given valid parenthesized infix arithmetic expression to postfix

expression. The expression consists of single character operands and the binary operators

```
+ (plus), - (minus), * (multiply) and / (divide)
#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>
#include <string.h>
#define MAX 50
char stack[MAX];
int top = -1;
void push(char c)
     if(top == MAX-1)
          printf("Stack overflow!!!");
     else
     {
          top++;
          stack[top] = c;
     }
}
char pop()
     char c;
     if(top == -1)
          return -1;
     else
     {
          c = stack[top];
          top--;
          return c;
     }
}
int isOperator(char c)
     if(c == '^' || c == '*' || c == '/' || c == '+' || c =='-')
          return 1;
     else
```

```
return 0;
}
int precedence(char c)
     if(c == '^')
          return 3;
     else if(c == '*' || c == '/')
          return 2;
     else if(c == '+' || c == '-')
          return 1;
     else
          return 0;
}
void convert(char infix[],char postfix[])
     int i=0, j=0;
     char item, x;
     push('(');
     strcat(infix,")");
     item = infix[i];
     while (item != ' \setminus 0')
          if(item == '(')
                push(item);
          else if(isalpha(item) || isdigit(item))
                postfix[j] = item;
                j++;
          else if(isOperator(item) == 1)
                x = pop();
                while(isOperator(x) == 1 && precedence(x) >=
precedence(item))
                     postfix[j] = x;
                     j++;
                     x = pop();
                push(x);
                push(item);
```

```
else if(item == ')')
               x = pop();
               while(x != '(')
                    postfix[j] = x;
                    j++;
                    x = pop();
               }
          }
          else
               printf("Invalid expression!\n");
               exit(0);
          i++;
          item = infix[i];
     }
     if(top > 0)
          printf("Invalid expression!!!\n");
          exit(0);
     postfix[j] = ' \0';
}
int main(int argc,char **argv)
     char infix[MAX],postfix[MAX];
     printf("Enter the infix expression: ");
     gets(infix);
     convert(infix,postfix);
     printf("\nYour postfix expression is:");
     puts(postfix);
     return 0;
}
```

INFIX TO POSTFIX

```
void push (chard)
       y (top == MAX-1)
Nintf (" Stack oungeon");
 ٤
         else
        { to p++;
          Stack [top] = c;
  3
         30P ()
  char
       chan e;
       (1- = = 9at) pc
             recess -1;
        else
           C - HEVER [+OP];
           top - - ;
           return c',
      }
 3
  int is operator (charc)
       4 ( c == 'A' || c == '* || ( == '/' || c == '+' || c == '-')
         return 1;
       else
         yeurn o;
   z
```

```
int precedence (charc)
4
    A (c .= , v)
        russy 3;
    esse if ( C == '+' 11 C == '1')
           yeuten 2;
    else of ( == '+' " c = = '-')
          YELEN DI
    ese russo o;
     convert ( char infi x[], char positive[])
    int 1:0, 1:0;
      char sam, +;
      push (. (.);
      streat ( infit , " )");
      item = infintj];
      white ( stem! = 1101)
         if ( item = = '(')

push ( item);
         esse if (indigit(item)11 isaspha (irem))
               postti * [j] = itemi,
                1++;
               y ( is Operator ( it em) = = 1)
           4
         esse
          while ( is operator (+) == 1 As precedence (+) > = pere custonce (+)
               postinci] = x;
                   j++:
                * = POP(),
              push (+);
              push ( Irem ;
         else ij ( item = = ')')
           1 pop (); while ( item ) = ( ( )
             f postfiv(i) = ";
                 " = bob();
              3
         3
         else
```

```
Enter the infix expression: a+b*c-f
```

Your postfix expression is:abc*+f-Press any key to continue . . .

Enter the infix expression: a+b-c^d

Your postfix expression is:ab+cd^-Press any key to continue . . .

Lab Program 3:

WAP to simulate the working of a queue of integers using an array. Provide the following operations

a) Insert b) Delete c) Display

The program should print appropriate messages for queue empty and queue overflow Conditions

```
#include <stdio.h>
#include <stdlib.h>
#define MAX 3
int front = -1;
int rear = -1;
int Q[MAX];
void enQ()
{
     int element;
    if(rear == MAX-1)
          printf("Q is full!\n");
     else
          if(front == -1 \&\& rear == -1)
                front = 0;
          rear++;
          printf("Enter element: ");
          scanf("%d", &element);
          Q[rear] = element;
     }
}
void deQ()
     if(rear == -1)
          printf("Q is empty!\n");
     else if(front == rear)
          printf("Element removed is: %d\n",Q[front]);
          front = rear = -1;
     else
```

```
{
          printf("Element removed is: %d\n",Q[front]);
          front++;
     }
}
void display()
    if(rear == -1)
          printf("Q is empty!\n");
     else
          for(int i=front;i<=rear;i++)</pre>
               printf("%d\t",Q[i]);
     printf("\n");
}
int main(int argc,char** argv)
    int choice;
    while(choice != 4)
            printf("----Q----\nEnter
choice:\n1.Insert\n2.Delete\n3.Display\n4.exit\n");
            scanf("%d",&choice);
            switch(choice)
                case 1:enQ();
                        break;
                 case 2:deQ();
                       break;
                 case 3:display();
                       break;
                 case 4:exit(0);
                default:exit(0);
            }
    return 0;
}
```

```
#include (stdio.h)
#include < stdUb. h>
#define SIZE 3
int front : -1;
Int reas : -1;
ine & [SIZE];
void Engint ?;
int Deals;
usid displayer;
int main (int argo, char * n argv)
   int choice, item;
   do
     prints ( . In Enter choice 1. End 2. Dea 3. distay 6. Ente: ");
      scant ("1.8, 1 choice);
      switch (choice)
         case 1: prints (" Enter the element to be added to the 9.");
                 Scanf ("1.d", & stem);
                Eng (item);
                break;
         case 2: item = Deg();
                4 ( item = = -1)
                    brink (" Q is empty");
                 else
                   printy ( " Irem removed from 10: 1.8", crem);
                break;
        case 3: display();
                 break;
        case 4: erit (0);
```

```
classmate
 } while (choice != 4);
 retion o;
void Englint el
if ( reax = = (SIZE-1))
 printy (" 9 is face");
else
if (reax == -1)
   front = = 0;
  rear + = 1:
 O [rear] = el;
int Ded ()
if ( front == -1)
  return -1;
 else
  item = Oltrone];
    if ( post rear)
    front = -1;
    2
```

```
void display()
     ( front = = -1)
     print f (" q is empty");
  erse
    prient f (" & contents are; ");
  for ( i = front ; i ( = rear ; i++)
    prints ("1.dit", g[i]);
```

```
Enter choice:
1.Insert
2.Delete
3.Display
4.exit
Enter element: 10
----0----
Enter choice:
1.Insert
2.Delete
3.Display
4.exit
Enter element: 20
----Q-----
Enter choice:
1.Insert
2.Delete
3.Display
4.exit
Enter element: 30
----0----
Enter choice:
1.Insert
2.Delete
3.Display
4.exit
Q is full!
----Q----
Enter choice:
1.Insert
2.Delete
3.Display
4.exit
10
       20
               30
----Q-----
Enter choice:
1.Insert
2.Delete
3.Display
4.exit
```

```
----Q-----
Enter choice:
1.Insert
2.Delete
3.Display
4.exit
Element removed is: 10
----Q-----
Enter choice:
1.Insert
2.Delete
3.Display
4.exit
Element removed is: 20
----Q-----
Enter choice:
1.Insert
2.Delete
3.Display
4.exit
```

LabProgram 4:

WAP to simulate the working of a circular queue of integers using an array. Provide the following operations.

a) Insert b) Delete c) Display

The program should print appropriate messages for queue empty and queue overflow Conditions

```
#include <stdio.h>
#include <stdlib.h>
#define MAX 3
int front = -1;
int rear = -1;
int Q[MAX];
void enQ()
{
     int element;
    if((front == 0 \&\& rear == MAX-1) \mid | (front == (rear+1) %MAX))
          printf("Q is full!\n");
     else
          if(front == -1 \&\& rear == -1)
                front = 0;
          rear = (rear+1)%MAX;
          printf("Enter element: ");
          scanf("%d", &element);
          Q[rear] = element;
     }
}
void deQ()
     if(rear == -1)
          printf("Q is empty!\n");
     else if(front == rear)
          printf("Element removed is: %d\n",Q[front]);
          front = rear = -1;
     else
```

```
{
          printf("Element removed is: %d\n",Q[front]);
          front = (front+1)%MAX;
     }
}
void display()
    if(rear == -1)
          printf("Q is empty!\n");
     else if(front <= rear)</pre>
           for(int i=front;i<=rear;i++)</pre>
                printf("%d\t",Q[i]);
          printf("\n");
     }
     else
           for(int i=front;i< MAX;i++)</pre>
                printf("%d\t",Q[i]);
           for(int i=0;i<= rear;i++)</pre>
                printf("%d\t",Q[i]);
          printf("\n");
     }
}
int main(int argc, char** argv)
    int choice;
    while (choice != 4)
             printf("----Circular Q----\nEnter
choice:\n1.Insert\n2.Delete\n3.Display\n4.exit\n");
             scanf("%d", &choice);
             switch(choice)
                 case 1:enQ();
                        break;
                 case 2:deQ();
                         break;
                 case 3:display();
                        break;
                 case 4:exit(0);
                 default:exit(0);
    return 0;
```

```
void end()
    int element;
    if ((front == 0 && reax == MAX-1) | (front == (sear+1) 1. MAX)
    prints ("Q is jue : In");
    erse
    if ( Front == -1 LE real == -1)
    front = 0;
     Year = (Yeax +1) 7. MAX;
     printf ("Enter element: ");
     scant ("1.d", Lelement);
   Q[reax] = element;
void de S()
   if (rear == -1)
    print ( " a is comptey: ");
   esse if ( grant = = rear)
```

```
Classmate
       print + (" Element removed is 1.410", Q[front]);
      tront + reax = -1:
  else
      print; ( " Element removed is 7.8 in", & ( pront);
      Mont = (Mont + 1) 1. MAX;
void display!)
    ij ( 100x = = -1)
         print f (" a is empty");
     esse if ( front < = reax)
            for (int i front; it = reax; i+1)
               print; ("T.dit", g(i));
            prints ("10");
    else
    ٤
         for ( sat ; short ; i < MAX ; i++)
              print; ("1.0/t", Q[i]);
         for (int i=0; it= reax; i++)
             print + ("7.012", Q (i));
         printf("In");
```

```
Enter choice:
1.Insert
2.Delete
3.Display
4.exit
Element removed is: 30
----Circular Q----
Enter choice:
1.Insert
2.Delete
3.Display
4.exit
Q is empty!
----Circular Q----
Enter choice:
1.Insert
2.Delete
3.Display
4.exit
```

LabProgram 5:

WAP to Implement Singly Linked List with following operations a) a) Create a linked list. b) Insertion of a node at first position, at any position and at end of

list. c) Display the contents of the linked list.

```
//Linked List 5
#include <stdio.h>
#include <stdlib.h>
struct node{
     int data;
     struct node *next;
};
void insertfirst(struct node **headptr) {
     struct node *newnode;
     int value;
     printf("Enter value: ");
     scanf("%d", &value);
    newnode = (struct node*)malloc(sizeof(struct node));
     newnode->data = value;
     newnode->next = NULL;
     if(*headptr == NULL)
          (*headptr) = newnode;
     else{
          newnode->next = (*headptr);
          (*headptr) = newnode;
     }
void insertpos(struct node **headptr) {
     struct node *newnode, *temp;
     int count=0, currpos=1, value, pos;
     printf("Enter value: ");
     scanf("%d", &value);
     newnode = (struct node*)malloc(sizeof(struct node));
     newnode->data = value;
     newnode->next = NULL;
     if((*headptr) == NULL){
        (*headptr) = newnode;
    }
    else{
          temp = (*headptr);
          while(temp != NULL) {
               count++;
               temp = temp->next;
          }
```

```
printf("There are %d elements in the list.Enter the
position where you want to insert the value: ", count);
          scanf("%d", &pos);
          if(pos > (count+1)){
               printf("no such position\n");
               return;
          if(pos == count+1) {
               temp = (*headptr);
               while(temp->next != NULL)
                    temp = temp->next;
               temp->next = newnode;
          }
          else{
               temp = (*headptr);
               while(temp->next != NULL) {
                     if(currpos == (pos-1)){
                          newnode->next = temp->next;
                          temp->next = newnode;
                         break;
                    currpos++;
                    temp = temp->next;
               }
          }
}
void insertlast(struct node **headptr) {
     struct node *newnode, *temp;
     int value;
     printf("Enter value: ");
     scanf("%d", &value);
    newnode = (struct node*)malloc(sizeof(struct node));
     newnode->data = value;
     newnode->next = NULL;
     temp = (*headptr);
     if(*headptr == NULL)
          (*headptr) = newnode;
     else{
          while(temp->next != NULL)
               temp = temp->next;
          temp->next = newnode;
void deletelast(struct node **headptr){
     struct node *temp;
     temp = (*headptr);
```

```
if((*headptr) == NULL)
          printf("The list is empty\n");
     else if((*headptr)->next == NULL)
          (*headptr) = NULL;
     else{
          temp = *headptr;
          while((temp->next)->next != NULL)
               temp = temp->next;
          temp->next = NULL;
     }
}
void display(struct node *temp) {
     if(temp == NULL){
          printf("The list is empty\n");
        return;
     }
     else{
          while(temp != NULL) {
               printf("%d\t", temp->data);
               temp = temp->next;
          printf("\n");
     }
}
int main(int argc,char **argv){
     int choice;
     struct node *head = NULL;
     while(choice != 6){
          printf("Enter choice 1)insertfirst 2)insertpos
3) insertlast 4) deletelast 5) display 6) exit: ");
          scanf("%d", &choice);
          switch(choice) {
               case 1:insertfirst(&head);break;
               case 2:insertpos(&head);break;
               case 3:insertlast(&head);break;
               case 4:deletelast(&head);break;
               case 5:display(head);break;
               case 6:
               default:exit(0);
          }
     }
     return 0;
}
```

	HL ALE	papergric
		Date: / /
	Lah Bogram 5	
1	struct reduct	
	int data;	
#	struct node enext;	
ļ	1;	
	f (trace assor struct of the approx)	
ŀ	wall rede + recorde;	
H	int value;	
+	printy ("Enter value; ");	
#	cang (" " " tvalue);	1990
#	nounous = (gruck node +) and loc (size of the	t mar)
4	ift votusi.	
\parallel	plinkt fo El	
4	neurode -> date = value;	
4	numede -> next = NULL;	
ļ	if (+ + codets = = mull)	
	(*headptr) = newcode;	
	else 2	
	nuonode > next = (+ headpts);	
	(+ wadpty) = newscode;	
	}	
	}	
	void insert pos (souch node ++ headpar) 1	
	waser node + newrocks, + temp;	
	int count = occurpos = 1, value, pos;	
	print ("Enter value: ");	
I	gent (" 1. d", braine);	
I	newhode = (smuch node +) madoc (esizee	y (sauce node));
I	neurode - date = value;	U
İ	newrode - next = NULL;	
1	(+ Lead DE == MOD NULL) }	
#	(* " and per) = numode;	

```
papergrid
                           Date: / /
clse ?
 ears = ( + hadptx);
 while (temp! = NULL) {
    temp = temp > next;
 }
 print ("There are "I.d chancals in the list. Enter position").
 scanf (" 1/2 d", & pos);
 if (pos > (vout+1)) &
     print ( No was position (");
 a return'
 if (PO) == count +1) {
    temp = (* headptr);
    while (tamp -) next ! = NULL)
     temp= temp = next;
   temp - next = number;
 else f
   temp = ( * headptr),
    white ( semp -> next! = NULL) }
     if (cumpos = = pos-1) &
       newnode -> next = temp -> next;
      temp - next = new node;
       break,
      Curron + -1;
      ltm p = temp + next;
```

```
papergrid
                                             Date: / /
void insert lade (struct node # + hadpt + ) }
   Struct nede + new node, + temp;
  int value;
  printy ( " # all value ");
   scart (" "1d", b value )",
  newscode = ( vivet node +) malloc ( size of ( sand node));
   newhoole -> data = value,
  remode - rext = NULL"
  timp = (+ neadpty);
  if ( * Leadpry = = NULL)
       (+ madptr) = numbele;
  else s
       ( LUN = 1 then equal) eight
           temp : trap + next;
      timp + next = numbedl;
  4
1
yord descretant (struct node to heapper)?
       quice node + exmp,
       semp = ( + headplx);
       if (+ headpts == NUCL)
        prints ( "The list is emply 'n")",
        esse is ((+ modpt+)- next == NULL).
           (+ meadpt 1) = NOLL'
       else 1
             temp > + head pri;
             while ( ( semp = next ) = next ! = NULL)
             temp : temp + next;
           tem po next = NULL.
```

```
Enter choice 1)insertfirst 2)insertpos 3)insertlast 4)deletelast 5)display 6)exit : 1
Enter value: 10
Enter choice 1)insertfirst 2)insertpos 3)insertlast 4)deletelast 5)display 6)exit : 2
There are 1 elements in the list.Enter the position where you want to insert the value: 2
Enter choice 1)insertfirst 2)insertpos 3)insertlast 4)deletelast 5)display 6)exit : 3
Enter value: 45
Enter choice 1)insertfirst 2)insertpos 3)insertlast 4)deletelast 5)display 6)exit : 5
10
       24
               45
Enter choice 1)insertfirst 2)insertpos 3)insertlast 4)deletelast 5)display 6)exit : 4
Enter choice 1)insertfirst 2)insertpos 3)insertlast 4)deletelast 5)display 6)exit : 5
10
       24
Enter choice 1)insertfirst 2)insertpos 3)insertlast 4)deletelast 5)display 6)exit :
Enter value: 2
Enter choice 1)insertfirst 2)insertpos 3)insertlast 4)deletelast 5)display 6)exit : 1
Enter value: 3
Enter choice 1)insertfirst 2)insertpos 3)insertlast 4)deletelast 5)display 6)exit : 3
Enter value: 5
Enter choice 1)insertfirst 2)insertpos 3)insertlast 4)deletelast 5)display 6)exit : 5
Enter choice 1)insertfirst 2)insertpos 3)insertlast 4)deletelast 5)display 6)exit : 2
Enter value: 32
There are 3 elements in the list.Enter the position where you want to insert the value: 2
Enter choice 1)insertfirst 2)insertpos 3)insertlast 4)deletelast 5)display 6)exit : 5
               2
Enter choice 1)insertfirst 2)insertpos 3)insertlast 4)deletelast 5)display 6)exit :
```

LabProgram 6:

WAP to Implement Singly Linked List with following operations

a) Create a linked list. b) Deletion of first element, specified element and last element in

the list. c) Display the contents of the linked list.

```
//Linked List 6
#include <stdio.h>
#include <stdlib.h>
struct node{
     int data;
     struct node *next;
};
void insertlast(struct node **headptr) {
     struct node *newnode, *temp;
     int value;
     printf("Enter value: ");
     scanf("%d", &value);
    newnode = (struct node*)malloc(sizeof(struct node));
     newnode->data = value;
     newnode->next = NULL;
     temp = (*headptr);
     if(*headptr == NULL)
          (*headptr) = newnode;
     else{
          while(temp->next != NULL)
               temp = temp->next;
          temp->next = newnode;
     }
void deletefirst(struct node **headptr) {
     if((*headptr) == NULL)
          printf("The list is empty\n");
     else if((*headptr)->next == NULL)
          (*headptr) = NULL;
     else{
          (*headptr) = (*headptr)->next;
     }
void deletepos(struct node **headptr) {
     struct node *temp;
     int count=0, currpos=1, pos;
     if((*headptr) == NULL){
```

```
printf("The list is empty\n");
          return;
     }
     else{
          if((*headptr)->next == NULL){
               (*headptr) = NULL;
               printf("The only element in the list was
deleted\n");
               return;
          }
        temp = (*headptr);
        while(temp != NULL) {
               count++;
               temp = temp->next;
          printf("There are %d elements in the list.Enter pos of
element to be deleted: ", count);
          scanf("%d", &pos);
        if(pos > (count+1)){
               printf("No such position is present\n");
               return;
          if(pos == (count+1)){
               temp = (*headptr);
               while((temp->next)->next != NULL)
                     temp = temp->next;
               temp->next = NULL;
          }
          else{
               temp = (*headptr);
               while(temp->next != NULL) {
                if(pos == 1){
                     (*headptr) = (*headptr)->next;
                    return;
                }
                     if(currpos == pos-1){
                          temp->next = (temp->next)->next;
                          return;
                     currpos++;
                     temp = temp->next;
               printf("No such element was found!!!\n");
          }
     }
void deletelast(struct node **headptr) {
```

```
struct node *temp;
     temp = (*headptr);
     if((*headptr) == NULL)
          printf("The list is empty\n");
     else if((*headptr)->next == NULL)
          (*headptr) = NULL;
     else{
          temp = *headptr;
          while((temp->next)->next != NULL)
               temp = temp->next;
          temp->next = NULL;
}
void display(struct node *temp) {
     if(temp == NULL) {
          printf("The list is empty\n");
        return;
     }
     else{
          while(temp != NULL) {
               printf("%d\t", temp->data);
               temp = temp->next;
          printf("\n");
     }
}
int main(int argc,char **argv){
     int choice;
     struct node *head = NULL;
     while(choice != 6){
          printf("Enter choice 1)insertlast 2)deletefirst
3) deletepos 4) deletelast 5) display 6) exit: ");
          scanf("%d", &choice);
          switch(choice) {
               case 1:insertlast(&head);break;
               case 2:deletefirst(&head);break;
               case 3:deletepos(&head);break;
               case 4:deletelast(&head);break;
               case 5:display(head);break;
               case 6:
               default:exit(0);
     return 0;
}
```

	med is	papergrid
		Date: / /
	6	
	cap program 6	
	# include (stdio.h)	
	# include (stallib.h.)	
	state node +	
	in data;	
	struct apple + next;	
	3.	
	void insertlant (water node ++ headspir) &	
	winter node + rewords, + temp,	
	int value;	
	prints 1" tales value: "",	
-	scant (" 1.3", &value");	
	new node = (Strengt node *) mallor ()izec	P (Vine 1 ros
	newword + data = velue;	
	punnade - next = NULL;	
	ump = (+hadpta);	
	is (* Lead ptx == NULL)	
	(theadptr) = newnode;	
11.11 12.21	Use 4	
	while (sup a next != NULL)	
	pund = dung ;	
	lemps next = numerale;	
	}	
	3	
	void descript (serve node + + head ptx) }	
	Hint to the same of the same o	
	if (* leadphy == NULL)	
	private ("The state to exceptly "r");	
	ese if ((+ headp+ 1) -> next = : NUL	2
	(+ read bis) = MARY;	()
	ene {	
	(+ Lead ptr) = (+madph) -> nex t;	
	}	
	3	

```
papergrid
                                             Date: / /
void deletepos (struct rade ++ headpts) }
  wind rade + temp;
  in court = ocumpor = 1, pos;
  if ((+headpro) = = NULL) 4
         print (" 11 to a empty ( ");
  7
  ese 1
       y (Hhead prr) + next == HULL) }
          ( + head pro) = NULL;
         return;
      eamp = ( + headptr);
      while (remp!= NULL) {
             count ++;
           tump = tump + next;
       prints (" There are yed eleventy. Ever position: ", court");
       scart (" y. d" , pos);
       if ( pos > count +1) }
           prints (" no war posición");
      if ( bos = = const.) }
            while ((temp- next) > next 1- NULL)
              camp : tamp + next;
           temp + next = NULL;
           temp = (the adpti);
           where comps next != NULL) }
                in ( pos = = 1 ) {
                    ex headpir) = (+ headpr) - next;
                    jetun;
```

```
papergrid
                                            Date: / /
   if (compos = = po) . 1) &
        sorp = next = (sump = next) = bext;
        3
       curros++;
       temp = temp > next;
     print ( " no such element found ( n");
void deletelant ( the is node ++headpry) {
        while rode + timp;
       temp = (* hadpty);
       if (c x proable) = = untr)
            prints (" list is emply , n');
      erce if (( * near exp+) - next := null) }
         ( theadpty) = NULL;
      2130 $
         temp = (+ headptx);
         while (George + next) + next- != NULL)
          timp: timp = next;
        puby vixt - NOTT!
      }
```

```
Enter choice 1)insertlast 2)deletefirst 3)deletepos 4)deletelast 5)display 6)exit : 1
Enter value: 10
Enter choice 1)insertlast 2)deletefirst 3)deletepos 4)deletelast 5)display 6)exit : 1
Enter value: 20
Enter choice 1)insertlast 2)deletefirst 3)deletepos 4)deletelast 5)display 6)exit : 1
Enter value: 30
Enter choice 1)insertlast 2)deletefirst 3)deletepos 4)deletelast 5)display 6)exit : 1
Enter value: 40
Enter choice 1)insertlast 2)deletefirst 3)deletepos 4)deletelast 5)display 6)exit : 5
               30
                       40
Enter choice 1)insertlast 2)deletefirst 3)deletepos 4)deletelast 5)display 6)exit : 2
Enter choice 1)insertlast 2)deletefirst 3)deletepos 4)deletelast 5)display 6)exit : 5
Enter choice 1)insertlast 2)deletefirst 3)deletepos 4)deletelast 5)display 6)exit : 4
Enter choice 1)insertlast 2)deletefirst 3)deletepos 4)deletelast 5)display 6)exit : 5
20
Enter choice 1)insertlast 2)deletefirst 3)deletepos 4)deletelast 5)display 6)exit : 3
There are 2 elements in the list.Enter pos of element to be deleted: 1
Enter choice 1)insertlast 2)deletefirst 3)deletepos 4)deletelast 5)display 6)exit : 5
30
Enter choice 1)insertlast 2)deletefirst 3)deletepos 4)deletelast 5)display 6)exit :
Enter choice 1)insertlast 2)deletefirst 3)deletepos 4)deletelast 5)display 6)exit : 1
Enter value: 1
Enter choice 1)insertlast 2)deletefirst 3)deletepos 4)deletelast 5)display 6)exit : 1
Enter value: 2
Enter choice 1)insertlast 2)deletefirst 3)deletepos 4)deletelast 5)display 6)exit : 1
Enter value: 3
Enter choice 1)insertlast 2)deletefirst 3)deletepos 4)deletelast 5)display 6)exit : 1
Enter value: 4
Enter choice 1)insertlast 2)deletefirst 3)deletepos 4)deletelast 5)display 6)exit : 5
       2
Enter choice 1)insertlast 2)deletefirst 3)deletepos 4)deletelast 5)display 6)exit : 2
Enter choice 1)insertlast 2)deletefirst 3)deletepos 4)deletelast 5)display 6)exit : 3
There are 3 elements in the list.Enter pos of element to be deleted: 6
No such position is present
Enter choice 1)insertlast 2)deletefirst 3)deletepos 4)deletelast 5)display 6)exit :
```

LabProgram 7:

WAP Implement Single Link List with following operations

a) Sort the linked list. b) Reverse the linked list. c) Concatenation of two linked lists

```
//Lab Program 7
//sort, reverse, concatenate linked lists(s)
#include <stdio.h>
#include <stdlib.h>
struct node{
     int data;
     struct node *next;
};
void insertend(struct node **headptr) {
     struct node *newnode, *temp;
     newnode = (struct node*)malloc(sizeof(struct node));
     int value;
     printf("Enter value: ");
     scanf("%d", &value);
     newnode->data = value;
     newnode->next = NULL;
     if((*headptr) == NULL)
          (*headptr) = newnode;
     else{
          temp = (*headptr);
          while(temp->next != NULL)
               temp = temp->next;
          temp->next = newnode;
     }
void deleteend(struct node **headptr) {
     struct node *temp;
     if((*headptr) == NULL)
          printf("The list is empty\n");
     else{
          temp = (*headptr);
          while((temp->next)->next != NULL)
               temp = temp->next;
          temp->next = NULL;
void display(struct node *temp) {
     if(temp == NULL)
```

```
printf("The list is empty\n");
     else{
          while(temp != NULL) {
               printf("%d\t", temp->data);
               temp = temp->next;
          printf("\n");
     }
void sort(struct node *temp) {
     struct node *p, *q;
     int a;
     for(p = temp;p != NULL;p = p->next) {
          for (q=p->next; q != NULL; q = q->next) {
                if(p->data > q->data){
                     a = p - > data;
                     p->data = q->data;
                     q->data = a;
                }
          }
     printf("The sorted list is as follows:\n");
     while(temp != NULL) {
          printf("%d\t", temp->data);
          temp = temp->next;
     printf("\n");
}
void reverse(struct node *temp) {
     struct node *first=NULL, *second, *third;
     second = temp;
     while(second != NULL) {
          third = second->next;
          second->next = first;
          first = second;
          second = third;
     temp = first;
     printf("The list after reversal is as follows:\n");
     while (temp != NULL) {
          printf("%d\t", temp->data);
          temp = temp->next;
     printf("\n");
void concatenate(struct node *temp1, struct node *temp2) {
     if(temp1 == NULL && temp2 == NULL) {
```

```
printf("Lists are empty\n");
    }
    else if(temp1 != NULL && temp2 == NULL) {
        while(temp1 != NULL) {
            printf("%d\t", temp1->data);
            temp1 = temp1->next;
        printf("\n");
    else if(temp1 == NULL && temp2 != NULL) {
        while(temp2 != NULL) {
            printf("%d\t", temp2->data);
            temp2 = temp2 -> next;
        printf("\n");
    }
    else{
        struct node *temp;
        temp = temp1;
        while(temp->next != NULL)
            temp = temp->next;
        temp->next = temp2;
        temp = temp1;
        printf("After concatenation:\n");
        while (temp != NULL) {
            printf("%d\t", temp->data);
            temp = temp->next;
        printf("\n");
    }
}
int main(int argc, char **argv) {
     int choice;
     struct node *head1 = NULL, *head2 = NULL;
     while (choice != 12) {
          printf("Enter choice : 1)insert1 2)insert2 3)delete1
4) delete2 5) display1 6) display2 7) sort1 8) sort2 9) reverse1
10) reverse2 11) concatenate 12) exit: ");
          scanf("%d", &choice);
          switch(choice) {
               case 1:insertend(&head1);break;
               case 2:insertend(&head2);break;
               case 3:deleteend(&head1);break;
               case 4:deleteend(&head2);break;
               case 5:display(head1);break;
               case 6:display(head2);break;
```

```
case 7:sort(head1);break;
case 8:sort(head2);break;
case 9:reverse(head1);break;
case 10:reverse(head2);break;
case 11:concatenate(head1,head2);break;
case 12:
    default:exit(0);
}
```

	papergric
Lah Program 7.	
+ (great about still people took	
ing (som p == NULL)	
esed	
while (temp : > NOLE) }	
print + (" 7.04", semp-)	later);
3	
bxut (, /u ,)	
}	
}	
3 (que a son ture (which) to 6 ion	
you co made *p, +q;	
in a;	
for (b: rew b; b : 4011; b = b)	next) d
for (a = p > next; a != NULL	19: 87 next) }
if (p + data > q → data)	ł
a=p+data;	
produte = q + dara;	
q → data = a;	
}	
3	
print (" The sorted live is as follow	w!:");
while (temp! = NOL) !	\.
print (" ". d ve", samp a dos	٥),
i Jean + dung : dung	
J	
Print ("\");	
3	

	papergric
	Date: / /
void reuse (white rode + temp) }	4 #1.41
your made + yingt = nucl, + second	, Third,
sound a from Di	
white (report 1 = NULE) 1	
with = second -s next	
record I next = Hist;	
Fixt = record;	
second = third;	
}	
samp - juist;	
prince (" April Receives: " (n");	
white (xmp! = NULL) {	
print & (" 1.84", tempodoa);	
tump = tump = next;	
4	
prents("\9")",	
3	
void concatenate (which node + temp!	View rode 4 tim
in which node + 1emp;	
trm P = tem P'i	
(surpagext 1: NULL)	
punp = punp = next;	
temp > next = Emp?;	
tem po templi,	
print (" Ala concertantion:");	
while (sump! = NULL) (
pints 1 " xd (+", sup-1 do	1- \'
toub = prub suext;	1
3	
breath ("/ Di);	
1	
3	

```
Enter choice : 1)insert1 2)insert2 3)delete1 4)delete2 5)displav1 6)displav2 7)sort1 8)sort2 9)reverse1 10)reverse2 11)concatenate 12)exit: 1
 nter choice : 1)insert1 2)insert2 3)delete1 4)delete2 5)displav1 6)displav2 7)sort1 8)sort2 9)reverse1 10)reverse2 11)concatenate 12)exit: 1
 nter value: 20
nter value: 10
nter value: 20
nter choice: 1)insert1 2)insert2 3)delete1 4)delete2 5)display1 6)display2 7)sort1 8)sort2 9)reverse1 10)reverse2 11)concatenate 12)exit: 1
 nter choice: 1)insert1 2)insert2 3)delete1 4)delete2 5)display1 6)display2 7)sort1 8)sort2 9)reverse1 10)reverse2 11)concatenate 12)exit: 1
nter value: 45
 nter choice :
nter value: 52
              1)insert1 2)insert2 3)delete1 4)delete2 5)display1 6)display2 7)sort1 8)sort2 9)reverse1 10)reverse2 11)concatenate 12)exit: 1
 nter choice : 1)insert1 2)insert2 3)delete1 4)delete2 5)display1 6)display2 7)sort1 8)sort2 9)reverse1 10)reverse2 11)concatenate 12)exit: 2
nter value: 1
 nter choice : 1)insert1 2)insert2 3)delete1 4)delete2 5)display1 6)display2 7)sort1 8)sort2 9)reverse1 10)reverse2 11)concatenate 12)exit: 2
nter value: 3
nter choice : 1)insert1 2)insert2 3)delete1 4)delete2 5)display1 6)display2 7)sort1 8)sort2 9)reverse1 10)reverse2 11)concatenate 12)exit: 2
 nter value: 2
nter choice : 1)insert1 2)insert2 3)delete1 4)delete2 5)display1 6)display2 7)sort1 8)sort2 9)reverse1 10)reverse2 11)concatenate 12)exit: 2
nter value: 4
 nter choice : 1)insert1 2)insert2 3)delete1 4)delete2 5)display1 6)display2 7)sort1 8)sort2 9)reverse1 10)reverse2 11)concatenate 12)exit: 5
 0 20 12 45 52
nter choice : 1)insert1 2)insert2 3)delete1 4)delete2 5)display1 6)display2 7)sort1 8)sort2 9)reverse1 10)reverse2 11)concatenate 12)exit: 6
 . 3 2 4
inter choice : 1)insert1 2)insert2 3)delete1 4)delete2 5)display1 6)display2 7)sort1 8)sort2 9)reverse1 10)reverse2 11)concatenate 12)exit: 7
2 3 4

There choice: 1)insert1 2)insert2 3)delete1 4)delete2 5)display1 6)display2 7)sort1 8)sort2 9)reverse1 10)reverse2 11)concatenate 12)exit: 9

he list after reversal is as follows:
2 45 20 12 10
 2 45 20 12 10
nter choice : 1)insert1 2)insert2 3)delete1 4)delete2 5)display1 6)display2 7)sort1 8)sort2 9)reverse1 10)reverse2 11)concatenate 12)exit: 10
he list after reversal is as follows:
4 3 2 1
Enter choice : 1)insert1 2)insert2 3)delete1 4)delete2 5)display1 6)display2 7)sort1 8)sort2 9)reverse1 10)reverse2 11)concatenate 12)exit: 11
Enter choice : 1)insert1 2)insert2 3)delete1 4)delete2 5)display1 6)display2 7)sort1 8)sort2 9)reverse1 10)reverse2 11)o
oncatenate 12)exit: 1
Enter value: 1
Enter choice : 1)insert1 2)insert2 3)delete1 4)delete2 5)display1 6)display2 7)sort1 8)sort2 9)reverse1 10)reverse2 11)o
oncatenate 12)exit: 1
Enter value: 2
Enter choice : 1)insert1 2)insert2 3)delete1 4)delete2 5)display1 6)display2 7)sort1 8)sort2 9)reverse1 10)reverse2 11)o
oncatenate 12)exit: 1
Enter value: 3
Enter choice : 1)insert1 2)insert2 3)delete1 4)delete2 5)display1 6)display2 7)sort1 8)sort2 9)reverse1 10)reverse2 11)d
oncatenate 12)exit: 2
Enter value: 1
Enter choice : 1)insert1 2)insert2 3)delete1 4)delete2 5)display1 6)display2 7)sort1 8)sort2 9)reverse1 10)reverse2 11)d
oncatenate 12)exit: 2
Enter value: 2
Enter choice : 1)insert1 2)insert2 3)delete1 4)delete2 5)display1 6)display2 7)sort1 8)sort2 9)reverse1 10)reverse2 11)d
oncatenate 12)exit: 11
After concatenation:
Enter choice : 1)insert1 2)insert2 3)delete1 4)delete2 5)display1 6)display2 7)sort1 8)sort2 9)reverse1 10)reverse2 11)d
oncatenate 12)exit:
```

LabProgram 8:

WAP to implement Stack & Queues using Linked Representation

```
//Lab Program 8
//stack and Queue implementation
#include <stdio.h>
#include <stdlib.h>
struct node{
     int data;
     struct node *next;
};
void insertend(struct node **headptr) {
     struct node *newnode, *temp;
     int value;
     printf("Enter value: ");
     scanf("%d", &value);
    newnode = (struct node*)malloc(sizeof(struct node));
     newnode->data = value;
     newnode->next = NULL;
     temp = (*headptr);
     if(*headptr == NULL)
          (*headptr) = newnode;
     else{
          while(temp->next != NULL)
               temp = temp->next;
          temp->next = newnode;
     }
void deletefront(struct node **headptr) {
     if((*headptr) == NULL)
          printf("The list is empty\n");
     else if((*headptr)->next == NULL)
          (*headptr) = NULL;
     else{
          (*headptr) = (*headptr)->next;
}
void deleteend(struct node **headptr) {
     struct node *temp;
     temp = (*headptr);
     if((*headptr) == NULL)
          printf("The list is empty\n");
     else if((*headptr)->next == NULL)
          (*headptr) = NULL;
     else{
          temp = *headptr;
```

```
while((temp->next)->next != NULL)
               temp = temp->next;
          temp->next = NULL;
     }
void display(struct node *temp) {
     if(temp == NULL) {
          printf("The list is empty\n");
        return;
     }
     else{
          while(temp != NULL) {
               printf("%d\t", temp->data);
               temp = temp->next;
          printf("\n");
     }
int main(int argc, char **argv) {
     struct node *head1=NULL, *head2=NULL;
     int choice;
     while (choice != 7) {
          printf("Enter choice: 1)pushstack 2)popstack
3)displaystack 4)EnQ 5)DeQ 6)displayQ 7)exit : ");
          scanf("%d", &choice);
          switch(choice) {
               case 1:insertend(&head1);break;
               case 2:deleteend(&head1);break;
               case 3:display(head1);break;
               case 4:insertend(&head2);break;
               case 5:deletefront(&head2);break;
               case 6:display(head2);break;
               case 7:
               default:exit(0);
     }
}
```

	male	papergrid
_	lab program 8	
	wid insert and (source mode ++ headptx) }	
	West rode + newrode, + oun p;	
	ant rake;	
	print ("Edit Yabre:");	
	econt (" 1.0", & rabue);	
	numode: (vinco rode +) massoe (sized (
	versuoge -> gap = x chre;	MICO ABDO)),
	respecte + next = NULL;	
	temp = (* head per);	,
	i) (+ headper = = NOLE)	
	(*headp+x) = newnode;	
	et/6 1	
	while (temp - next) = NOLL)	
	sout = sout - vexc;	
	} temp > next = newwords;	
	}	
	V==4	
	void deservations (went node the redpts)	
	if ((* products) = = NOTT)	
	punt (" Lit 21 empty \n"),	
	cise if (c + madpti) + next = = null)	
	(*Madpir) = NULL;	
_	6176 \$	
_	(+ asadp +) = (+ headp++) + next;	
_	3	
_	3	
-	void duete and (water node ++ headpril	61.6 1
	cure about + comp;	an p = (+headph
_	emp = (Headpt);	wile ((Bump + next) +
_	i) ((+ headptr) = = HOLL)	mube enub yes
	prints (" List empty (")")	pub -> vexp -> vex
	che if ((+hiadpr) = nex t == NOL)	3

```
Enter choice: 1)pushstack 2)popstack 3)displaystack 4)EnQ 5)DeQ 6)displayQ 7)exit : 1
Enter value: 1
Enter choice: 1)pushstack 2)popstack 3)displaystack 4)EnQ 5)DeQ 6)displayQ 7)exit : 1
Enter value: 2
Enter choice: 1)pushstack 2)popstack 3)displaystack 4)EnQ 5)DeQ 6)displayQ 7)exit : 1
Enter value: 3
Enter choice: 1)pushstack 2)popstack 3)displaystack 4)EnQ 5)DeQ 6)displayQ 7)exit : 1
Enter value: 4
Enter choice: 1)pushstack 2)popstack 3)displaystack 4)EnQ 5)DeQ 6)displayQ 7)exit : 3
       2
Enter choice: 1)pushstack 2)popstack 3)displaystack 4)EnQ 5)DeQ 6)displayQ 7)exit : 2
Enter choice: 1)pushstack 2)popstack 3)displaystack 4)EnQ 5)DeQ 6)displayQ 7)exit : 3
Enter choice: 1)pushstack 2)popstack 3)displaystack 4)EnO 5)DeO 6)displayO 7)exit :
Enter choice: 1)pushstack 2)popstack 3)displaystack 4)EnQ 5)DeQ 6)displayQ 7)exit : 4
Enter value: 10
Enter choice: 1)pushstack 2)popstack 3)displaystack 4)EnQ 5)DeQ 6)displayQ 7)exit : 4
Enter value: 20
Enter choice: 1)pushstack 2)popstack 3)displaystack 4)EnQ 5)DeQ 6)displayQ 7)exit : 4
Enter value: 30
Enter choice: 1)pushstack 2)popstack 3)displaystack 4)EnQ 5)DeQ 6)displayQ 7)exit : 4
Enter value: 40
Enter choice: 1)pushstack 2)popstack 3)displaystack 4)EnQ 5)DeQ 6)displayQ 7)exit : 6
        20
Enter choice: 1)pushstack 2)popstack 3)displaystack 4)EnQ 5)DeQ 6)displayQ 7)exit : 5
Enter choice: 1)pushstack 2)popstack 3)displaystack 4)EnQ 5)DeQ 6)displayQ 7)exit : 6
Enter choice: 1)pushstack 2)popstack 3)displaystack 4)EnQ 5)DeQ 6)displayQ 7)exit :
```

LabProgram 9:

WAP Implement doubly link list with primitive operations

- a) Create a doubly linked list. b) Insert a new node to the left of the node.
- c) Delete the node based on a specific value. c) Display the contents of the list

```
//Lab Program 9
//insert, delete, display
#include <stdio.h>
#include <stdlib.h>
struct node{
     int data;
     struct node *next, *prev;
};
void insertend(struct node **headptr) {
     struct node *newnode, *temp;
     int value;
     printf("Enter value: ");
     scanf("%d", &value);
     newnode = (struct node*)malloc(sizeof(struct node));
     newnode->data = value;
     newnode->prev = NULL;
     newnode->next = NULL;
     if((*headptr) == NULL){
          (*headptr) = newnode;
     }
     else{
          temp = (*headptr);
          while(temp->next != NULL)
               temp = temp->next;
          temp->next = newnode;
          newnode->prev = temp;
void insertbefore(struct node **headptr) {
     struct node *newnode, *temp;
     int value, ele;
     printf("Enter value: ");
     scanf("%d", &value);
     newnode = (struct node*)malloc(sizeof(struct node));
     newnode->data = value;
     newnode->next = NULL;
```

```
newnode->prev = NULL;
     if((*headptr) == NULL)
          (*headptr) = newnode;
     else{
          printf("Enter the element before which value is to be
inserted: ");
          scanf("%d", &ele);
        if((*headptr)->data == ele){
            newnode->next = (*headptr);
            (*headptr)->prev = newnode;
            (*headptr) = newnode;
            return;
          temp = (*headptr);
          while(temp->next != NULL) {
               if((temp->next)->data == ele){
                    newnode->next = temp->next;
                     (temp->next) ->prev = newnode;
                    temp->next = newnode;
                    newnode->prev = temp;
                    return;
               temp = temp->next;
          printf("No such element found!\n");
}
void deleteend(struct node **headptr) {
     struct node *temp;
     if((*headptr) == NULL)
          printf("List is empty\n");
     else{
          temp = (*headptr);
          while((temp->next)->next != NULL)
               temp = temp->next;
          temp->next = NULL;
void deleteval(struct node **headptr){
     struct node *temp;
     int value;
     if((*headptr) == NULL)
          printf("List is empty\n");
     else{
          printf("Enter the value to be deleted: ");
          scanf("%d", &value);
        if((*headptr)->data == value){
```

```
(*headptr) = (*headptr)->next;
            (*headptr)->prev = NULL;
            return;
        temp = (*headptr);
        while((temp->next)->next != NULL)
            temp = temp->next;
        if((temp->next)->data == value){
            temp->next = NULL;
            return;
        }
          temp = (*headptr);
          while(temp->next != NULL) {
               if((temp->next)->data == value){
                     temp->next = (temp->next)->next;
                     (temp->next)->prev = temp;
                     return;
               temp = temp->next;
          printf("No such element found!\n");
     }
void display(struct node *temp) {
     if(temp == NULL)
          printf("List is empty\n");
     else{
          while(temp != NULL) {
               printf("%d\t", temp->data);
               temp = temp->next;
          printf("\n");
     }
}
int main(int charc, char **argv) {
     int choice;
     struct node *head=NULL;
     while (choice != 6) {
          printf("Enter choice: 1)insertend 2)insertbefore
3) deleteend 4) deleteval 5) display 6) exit : ");
          scanf("%d", &choice);
          switch(choice) {
               case 1:insertend(&head);break;
               case 2:insertbefore(&head);break;
               case 3:deleteend(&head);break;
               case 4:deleteval(&head);break;
```

	mars	papergric
Lab Pxgram 9:		
#include < state. h >		
#include (4deibh)		
and rode 1		
int data;		
give node *rext, *f	orev '.	
4:	,	
void insert poloe lastice was	6 4+wadpt) }	
church node + new rode , +		
int value, ele;		
frint (" Enter value : "	('w)'.	
scant (" ". d", scame		
newhoole = 1 struct node		(vauet Lode)
newhode -> data = value	•	
newprode - next = NO+	,	
nunted + prev = Nu		
if ((head prx) = = NO L.		
(thendpty) = nho		
else 4		
print + (" Enter the	element botore was	th vaclue is insu
scart 1" y.d ", 1 ele		
if ((Aproper) + a	95.	
	xt = (+ head por);	
(+ hudpir) -	prex = new node ;	
(decadper) =	nuonode;	
reitin:		
}	-	
i (rtg backs) = q mit		
white (temp + next) = N		
of (Genpanext)	dota = = ese) (
	xt = temp -> nac	τ;
(samp-next) -	pio = ne node	;
sen p -1 ne.	prev = tump;	
return;	1 m - Prush 1	

```
papergrid
                                              Date: / /
  temp : exmps next;
prints 1" No such demant found (n");
void deserval ( struct node ++ headper) !
     time made & temp;
     int value;
    if ((*headpr) = = NOLL)
     prints ("Det empry 10");
      Print ("Fater value to be deleted ; " )"
       scant ( " + & " . Lyous);
       1 ( une == alob = ( v+q band =) ji
           (*beadpr) = (*meadpr) - next;
          ( * wead ptr) => preu = NULL;
       tomp = ( + headper);
       while ((exemponent) + next ) = NOLL)
           long = temp+ next;
        if ( fem p -> next) -> data = = ratual 1
        temp = (+ he adptx);
        while ( lemp+ next ! = NULL) {
         if ((exceps next) + dosa = + value) }
             (mub + vext) = (mub + vext) + vext,
              (sump - next) > pred = temp;
          } tump = oumps next;
       prior + (" No such escent found (n');
```

```
Enter choice: 1)insertend 2)insertbetore 3)deleteend 4)deleteval 5)display b)exit :
Enter value: 10
Enter choice: 1)insertend 2)insertbefore 3)deleteend 4)deleteval 5)display 6)exit : 1
Enter value: 20
Enter choice: 1)insertend 2)insertbefore 3)deleteend 4)deleteval 5)display 6)exit : 1
Enter value: 30
Enter choice: 1)insertend 2)insertbefore 3)deleteend 4)deleteval 5)display 6)exit : 5
Enter choice: 1)insertend 2)insertbefore 3)deleteend 4)deleteval 5)display 6)exit : 2
Enter value: 15
Enter the element before which value is to be inserted: 20
Enter choice: 1)insertend 2)insertbefore 3)deleteend 4)deleteval 5)display 6)exit : 5
10
                20
Enter choice: 1)insertend 2)insertbefore 3)deleteend 4)deleteval 5)display 6)exit : 3
Enter choice: 1)insertend 2)insertbefore 3)deleteend 4)deleteval 5)display 6)exit : 5
10
                20
Enter choice: 1)insertend 2)insertbefore 3)deleteend 4)deleteval 5)display 6)exit : 4
Enter the value to be deleted: 10
Enter choice: 1)insertend 2)insertbefore 3)deleteend 4)deleteval 5)display 6)exit : 5
15
        20
Enter choice: 1)insertend 2)insertbefore 3)deleteend 4)deleteval 5)display 6)exit :
Enter choice: 1)insertend 2)insertbefore 3)deleteend 4)deleteval 5)display 6)exit : 1
Enter value: 10
Enter choice: 1)insertend 2)insertbefore 3)deleteend 4)deleteval 5)display 6)exit : 3
Enter choice: 1)insertend 2)insertbefore 3)deleteend 4)deleteval 5)display 6)exit : 5
List is empty
Enter choice: 1)insertend 2)insertbefore 3)deleteend 4)deleteval 5)display 6)exit : 1
Enter value: 10
Enter choice: 1)insertend 2)insertbefore 3)deleteend 4)deleteval 5)display 6)exit : 1
Enter value: 20
Enter choice: 1)insertend 2)insertbefore 3)deleteend 4)deleteval 5)display 6)exit : 1
Enter value: 30
Enter choice: 1)insertend 2)insertbefore 3)deleteend 4)deleteval 5)display 6)exit : 1
Enter value: 45
Enter choice: 1)insertend 2)insertbefore 3)deleteend 4)deleteval 5)display 6)exit : 2
Enter value: 42
Enter the element before which value is to be inserted: 45
Enter choice: 1)insertend 2)insertbefore 3)deleteend 4)deleteval 5)display 6)exit : 5
               30
                     42
       20
                             45
Enter choice: 1)insertend 2)insertbefore 3)deleteend 4)deleteval 5)display 6)exit :
```

LabProgram 10:

Write a program

- a) To construct a binary Search tree.
- b) To traverse the tree using all the methods i.e., in-order, preorder and post order
- c) To display the elements in the tree.

```
//Lab Program 10
//Binary search tree
#include <stdio.h>
#include <stdlib.h>
typedef struct BST{
     int data;
     struct BST *left, *right;
} node;
node *create() {
    node *newnode;
     int value;
     printf("Enter value: ");
     scanf("%d", &value);
     newnode = (node*)malloc(sizeof(node));
     newnode->data = value;
     newnode->left = NULL;
     newnode->right = NULL;
     return newnode;
void insert(node *root, node *temp) {
     if(temp->data < root->data) {
          if(root->left == NULL)
               root->left = temp;
          else{
               insert(root->left,temp);
     if(temp->data > root->data) {
          if(root->right == NULL)
               root->right = temp;
          else{
               insert(root->right, temp);
     }
```

```
void inorder(node *root) {
     if(root != NULL) {
          inorder(root->left);
          printf("%d\t", root->data);
          inorder(root->right);
     }
}
void preorder(node *root) {
     if(root != NULL) {
          printf("%d\t", root->data);
          preorder(root->left);
          preorder(root->right);
     }
}
void postorder(node *root) {
     if(root != NULL) {
          postorder(root->left);
          postorder(root->right);
          printf("%d\t", root->data);
     }
//node *minvaluenode(){}
//node *deletenode(){}
int main(int argc,char **argv){
     int choice;
     node *root=NULL, *temp;
     while (choice != 5) {
          printf("\nEnter choice 1)insert 2)inorder 3)preorder
4) postorder 5) exit : ");
          scanf("%d", &choice);
          switch(choice) {
               case 1:temp = create();
                     if(root == NULL)
                         root = temp;
                     else
                         insert(root, temp);
                     break;
               case 2:inorder(root);break;
               case 3:preorder(root);break;
               case 4:postorder(root);break;
               case 5:
               default:exit(0);
     return 0;
}
```

	media
	papergrid
	Date: / /
-	
	Lab Program 10:
	# include < yolio.hz
-	# I APRILITE STATE OF THE PARTY
	# increase < years by
_	1
	typeday wince ased
	int data;
	would by t bight, bright;
	3 node;
	rode +cuate() i
	rode +temp: 2 ne rople;
	ent value;
	pratt ("Entit value:");
	scant "" y. din", & ratue);
-	numnade = (struct node +) malloc (size of (node));
	new mode -> data = value;
	newhode -> left = hownode -> Kight: NULL;
	}
	roid inject (node & root, node & simp) (
	if (imprava + noor + data) t
	if (ropt + left = = HULL)
	toot -> left = temp;
	else
	invest (xoot > eath), temp);
	111-3411
	i) (samp + data > poor + data) {
	4 (100+ - 1/961 :: NOLL)
	(A) (S) NOLL)
	topt suidret = MATT,
	£150
	intert (not - sight, samp);
	1
	1

	papergrid
	Date: / /
biology (node + 1001)	
ig (1000 + 1 = NOLL) {	
inordu (root - left);	
prints ("1.0", root -> down);	
inordu (root -> rique);	
3	
upid preorder (node troot) (
if (1001 1 = NULL) 1	
presider	
print & (" 11.0", not + dota);	
preordy (root + light);	
preordu (root - right);	
}	
void portorder (no de + wat)	
ig (1001 1= NUL e) {	
portorder (real -> left);	
powordu (root -> right);	
prints (" Y.d" rootsdare);	
1	
\$	

```
Enter choice 1)insert 2)inorder 3)preorder 4)postorder 5)exit : 1
Enter value: 10
Enter choice 1)insert 2)inorder 3)preorder 4)postorder 5)exit : 1
Enter value: 2
Enter choice 1)insert 2)inorder 3)preorder 4)postorder 5)exit : 1
Enter value: 3
Enter choice 1)insert 2)inorder 3)preorder 4)postorder 5)exit : 1
Enter value: 7
Enter choice 1)insert 2)inorder 3)preorder 4)postorder 5)exit : 1
Enter value: 4
Enter choice 1)insert 2)inorder 3)preorder 4)postorder 5)exit : 2
                                10
               4
Enter choice 1)insert 2)inorder 3)preorder 4)postorder 5)exit : 3
Enter choice 1)insert 2)inorder 3)preorder 4)postorder 5)exit : 4
                       2
                                10
Enter choice 1)insert 2)inorder 3)preorder 4)postorder 5)exit :
Enter choice 1)insert 2)inorder 3)preorder 4)postorder 5)exit : 1
Enter value: 7
Enter choice 1)insert 2)inorder 3)preorder 4)postorder 5)exit : 1
Enter value: 3
Enter choice 1)insert 2)inorder 3)preorder 4)postorder 5)exit : 1
Enter value: 5
Enter choice 1)insert 2)inorder 3)preorder 4)postorder 5)exit : 1
Enter value: 45
Enter choice 1)insert 2)inorder 3)preorder 4)postorder 5)exit : 1
Enter value: 28
Enter choice 1)insert 2)inorder 3)preorder 4)postorder 5)exit : 2
                        28
                                45
Enter choice 1)insert 2)inorder 3)preorder 4)postorder 5)exit : 3
                        45
                                28
Enter choice 1)insert 2)inorder 3)preorder 4)postorder 5)exit : 4
                28
                        45
Enter choice 1)insert 2)inorder 3)preorder 4)postorder 5)exit :
```