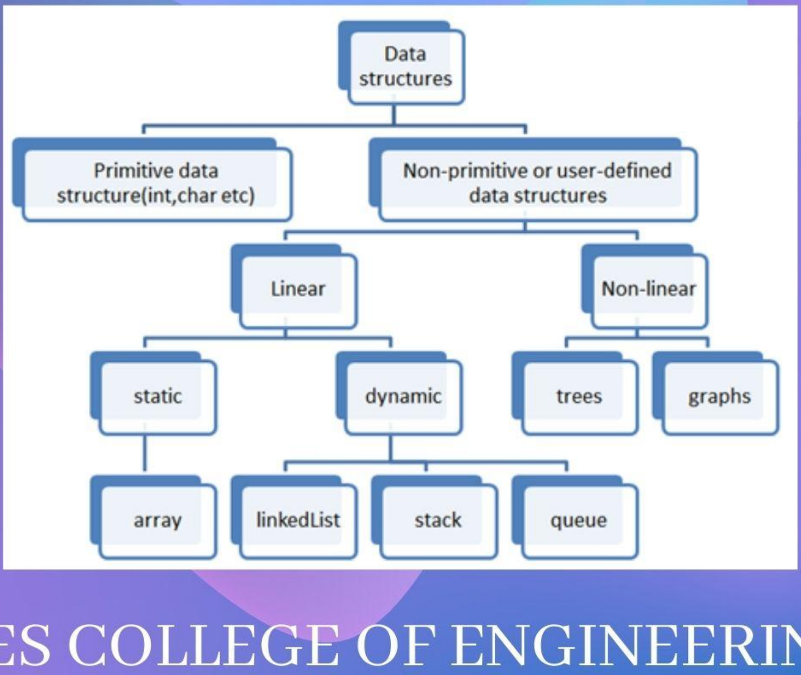
DEPARTMENT OF COMPUTER SCIENCE

Data Structures Lab Manual-P21CSL306


Data structures
Primitive data structure(int,char etc)
Non-primitive or user-defined data structures
Linear
Non-linear
static
dynamic
trees
graphs
array
linkedList
stack
queue

PES COLLEGE OF ENGINEERING

MANDYA-571401

Data Structures Laboratory [As per Choice Based Credit System (CBCS) & OBE Scheme] **SEMESTER – III**			
Course Code:	**P21CSL306**	**Credits:**	**01**
Teaching Hours/Week (L:T:P):	**0:0:2**	**CIE Marks:**	**50**
Total Number of Teaching Hours:	**24**	**SEE Marks:**	**50**
Note: All programs are to be implemented using C Language			

1.	Create a structure **DISTANCE** with data members ***kms*** and ***meters*** of type integer. Implement a program to perform addition and subtraction on two distances by passing pointer to a structure to function.
2.	Implement a menu driven program to perform the following operations on Singly Linked List. (i) Create SLL of 'n' nodes of integers (insert front/rear) (ii) Delete the node with specified integer from the list with appropriate message. (iii) Display the contents of the SLL.
3.	Implement a menu driven Program for the following operations on Doubly Linked List (DLL) of Library Data with the fields: BOOK_ID, BOOK_TITLE, AUTHOR, EDITION (i) Create a DLL of 'N' books (Insert front/rear). (ii) Count the number of nodes in the DLL. (iii) Delete the node at front/rear. (iv) Display the contents of DLL.
4.	Implement a menu driven Program for the following operations on Circular Linked List. (i) Create CLL of 'n' nodes of string. (insert front/rear) (ii) Count the number of nodes in the CLL. (iii) Delete the node at front/rear. (iv) Display the contents of CLL.
5.	Implement a menu driven Program for the following operations on STACK of Integers (Array Implementation of Stack with maximum size MAX) (i) Push an Element on to Stack (Handle the situation of overflow) (ii) Pop an Element from Stack (Handle the situation of underflow) (iii) Display the contents of Stack
6.	Implement a Program to convert a given infix expression to postfix expression.
7.	Implement the following using recursion: (i) Tower_of_Hanoi (ii) GCD of two numbers (iii) Largest of 'n' numbers
8.	Implement a menu driven Program for the following operations on QUEUES of Strings using Linked list (i) Insert an Element into Queue (ii) Delete an Element from Queue (iii) Display the contents of Queue
9.	Implement a menu driven program to perform the following operations on priority queue

	using linked list. (i) Insert a node based on priority. (ii) Delete a node from the queue (iii) Display the contents of the queue
10.	Implement a menu driven Program for the following operations on Binary Search Tree (BST) of Integers (i) Create a BST of N Integers (ii) Traverse the BST in Inorder, Preorder and Postorder

COs	**Statement**	**PO 1**	**PO 2**	**PO 3**	**PO 4**	**PO 5**	**PO 6**	**PO 7**	**PO 8**	**PO 9**	**PO 10**	**PO 11**	**PO 12**
CO1	**Design** algorithms using different data structures like List, Stack, Queue and Trees.	2	2	2									
CO2	**Develop** programs with suitable data structure based on the requirements of the real-time applications.	2	2	2									1

PROGRAM 1

Create a structure DISTANCE with data members kms and meters of type integer. Implement a program to perform addition and subtraction on two distances by passing pointer to a structure to function.

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

/* Structure Declaration*/
struct distance
{
        int kms;
        int meters;
};
typedef struct distance DISTANCE;

/*Function prototypes*/
DISTANCE add_distance (DISTANCE *, DISTANCE *);
DISTANCE subtract_distance (DISTANCE *, DISTANCE *);

/*Structure variable declaration*/
DISTANCE d1, d2, d3, d4;


int main()
{
         int option;
         clrscr();
        do
        {
                printf("\n ******** MAIN MENU *********");
                printf("\n 1. Read the distances ");
                printf("\n 2. Display the distances");
                printf("\n 3. Add the distances");
                printf("\n 4. Subtract the distances");
                printf("\n 5. EXIT");
                printf("\n Enter your option : ");
                scanf("%d", &option);
                switch(option)
                {
                        case 1:
                                printf("\n Enter the first distance  in kms and meters: ");
                                scanf("%d %d", &d1.kms, &d1.meters);
                                printf("\n Enter the second distance  in kms and meters: ");
                                scanf("%d %d", &d2.kms, &d2.meters);
                                break;
                        case 2:
                                printf("\n The first distance is : %d kms %d meters",d1.kms,
                                        d1.meters);
```

```
                    printf("\n The second distance is : %d kms %d
                            meters",d2.kms, d2.meters);
                    break;
                case 3:
                    d3 = add_distance(&d1, &d2);
                    printf("\n The sum of two distances is : %d kms %d meters",
                           d3.kms, d3.meters);
                    break;
                case 4:
                    d4 = subtract_distance(&d1, &d2);
                    printf("\n The difference between two distances is : %d kms
                           %d meters", d4.kms, d4.meters);
                    break;
            }
        }while(option != 5);
        getch();
        return 0;
}
```

/*function to add two distance*/

```
DISTANCE add_distance(DISTANCE *d1, DISTANCE *d2)
{
        DISTANCE sum;
        sum.meters = d1->meters + d2->meters;
        sum.kms = d1->kms + d2->kms;
        while (sum.meters >= 1000)
        {
                sum.meters = sum.meters % 1000;
                sum.kms += 1;
        }
        return sum;
}
```

/*function to subtract two distance*/

```
DISTANCE subtract_distance(DISTANCE *d1, DISTANCE *d2)
{
        DISTANCE sub;
        if(d1->kms > d2->kms)
        {
                sub.meters = d1->meters - d2->meters;
                sub.kms = d1->kms - d2->kms;
        }
        else
        {
                sub.meters = d2->meters - d1->meters;
                sub.kms = d2->kms - d1->kms;
        }
        if(sub.meters < 0)
        {
                sub.meters = sub.meters + 1000;
```

```
                    sub.kms -= 1;
          }
          return sub;
}
```

```
*********************************OUTPUT*********************************
 ******** MAIN MENU *********
 1. Read the distances
 2. Display the distances
 3. Add the distances
 4. Subtract the distances
 5. EXIT
 Enter your option : 1

 Enter the first distance  in kms and meters: 5 300

 Enter the second distance  in kms and meters: 3 400

 ******** MAIN MENU *********
 1. Read the distances
 2. Display the distances
 3. Add the distances
 4. Subtract the distances
 5. EXIT
 Enter your option : 2

 The first distance is : 5 kms 300 meters
 The second distance is : 3 kms 400 meters
 ******** MAIN MENU *********
 1. Read the distances
 2. Display the distances
 3. Add the distances
 4. Subtract the distances
 5. EXIT
 Enter your option : 3

 The sum of two distances is : 8 kms 700 meters
 ******** MAIN MENU *********
 1. Read the distances
 2. Display the distances
 3. Add the distances
 4. Subtract the distances
 5. EXIT
 Enter your option : 4

 The difference between two distances is : 1 kms 900 meters
 ******** MAIN MENU *********
 1. Read the distances
```

```
2. Display the distances
3. Add the distances
4. Subtract the distances
5. EXIT
Enter your option : 5
```

PROGRAM 2

Implement SLL which performs the following operations.

(i) Create SLL of ‘n’ integers(insert front/rear)
(ii) Delete the specified integer from the list with appropriate message.
(iii) Display the contents of the list.

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

/*Structure declaration for a node*/
struct node
{
	int info;
	struct node *link;
};

typedef struct node *NODE;


/*function to allocate memory for a node*/
NODE getnode(void)
{
	NODE x;
	x=(NODE)malloc(sizeof(struct node));
	if(x==NULL)
	{
		printf("\nOut of memory");
		exit(0);
	}
	return x;
}


/*function to insert a node at the rear end in SLL*/
NODE insert_rear(int item,NODE first)
{
	NODE temp,cur;
	temp=getnode();
	temp->info=item;
	temp->link=NULL;
	if(first==NULL)
		return temp;
	cur=first;
	while(cur->link!=NULL)
	{
		cur=cur->link;
	}
	cur->link=temp;
	return first;
}
```

/*function to delete a node whose is specified in SLL */

```
NODE delete_item(NODE first, int item)
{
	NODE prev, cur;
	if(first==NULL)
	{
		printf("\n List is Empty");
		return first;
	}
	/*Search for the item specified in each node*/
	prev=NULL;
	cur=first;
	while(cur!=NULL && item!=cur->info)
	{
		prev=cur;
		cur=cur->link;
	}
	if(cur==NULL)
	{
		printf("\n The item to be deleted is not found in the list");
		return first;
	}

	/*delete the node*/
	prev->link=cur->link;
	free(cur);
	return first;
}
```

/*function to display contents of the SLL*/

```
void display(NODE first)
{
	NODE temp;
	if(first==NULL)
	{
		printf("\nList is empty");
		return;
	}
	printf("\nThe contents of the list:");
	temp=first;
	while(temp!=NULL)
	{
		printf("%d\t",temp->info);
		temp=temp->link;
	}
```

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



int main()
{
        NODE first=NULL;
        int choice,item,n,i;
        //clrscr();
        do
        {
                printf("\n 1: Create List\n 2: Delete specified item\n 3: Display\n 4: Exit\n");
                printf("\nEnter the choice:");
                scanf("%d",&choice);
                switch(choice)
                {
                        case 1: printf("\nEnter the no. of nodes to create List:");
                                scanf("%d",&n);
                                for(i=1;i<=n;i++)
                                {
                                        printf("\nEnter the item %d to be inserted:",i);
                                        scanf("%d",&item);
                                        first=insert_rear(item,first);
                                   }
                                break;

                        case 2: printf("\nEnter the item to be deleted:");
                                 scanf("%d",&item);
                                first=delete_item(first,item);
                                 break;

                        case 3: display(first);
                                break;

                }
        }while(choice!=4);
        getch();
        return 0;
}
```
************************************OUTPUT********************************

```
 1:Create List
 2:Delete specified item
 3:Display
 4:Exit

Enter the choice:1
Enter the no. of nodes to create List:5
Enter the item 1 to be inserted:10
Enter the item 2 to be inserted:100
Enter the item 3 to be inserted:20
Enter the item 4 to be inserted:200
Enter the item 5 to be inserted:5

 1:Create List
 2:Delete specified item
 3:Display
 4:Exit

Enter the choice:3
The contents of the list: 10     100      20        200      5

 1: Create List
 2: Delete specified item
 3: Display
 4: Exit

Enter the choice:2
Enter the item to be deleted:6

 The item to be deleted is not found in the list
 1:Create List
 2:Delete specified item
 3:Display
 4:Exit

Enter the choice:2
Enter the item to be deleted:100

 1:Create List
 2:Delete specified item
 3:Display
 4:Exit

Enter the choice:3
The contents of the list:10      20        200      5

 1:Create List
 2:Delete specified item
 3:Display
 4:Exit

Enter the choice:4
```

PROGRAM 3

Implement a menu driven Program for the following operations on Doubly Linked List (DLL) of Library Data with the fields: BOOK_ID, BOOK_TITLE, AUTHOR, EDITION

(i) **Create a DLL of 'N' books (Insert front/rear).**
(ii) **Count the number of nodes in the DLL.**
(iii) **Delete the node at front/rear.**
(iv) **Display the contents of DLL**.

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

/*Structure Declaration*/
struct node
{
   int BOOK_ID;
   char BOOK_TITLE[20];
   char AUTHOR[30];
   char EDITION[3];
   struct node *llink;
   struct node *rlink;
};
typedef struct node *NODE;

/*function to allocate memory for a node*/
NODE getnode(void)
{
   NODE x;
   x=(NODE)malloc(sizeof(struct node));
   if(x==NULL)
   {
      printf("Out of memory..\n");
      exit(0);
   }
   return x;
}

/*function to insert a node into DLL at the rear-end*/
NODE insert_rear(NODE first)
{
   NODE temp,cur;
   temp=getnode();
   //temp->rlink=temp->llink=NULL;
   printf("\nEnter the BOOK_ID:");
   scanf("%d",&temp->BOOK_ID);
   printf("\nEnter the BOOK_TITLE:");
   scanf("%s",temp->BOOK_TITLE);
   printf("\nEnter the AUTHOR:");
   scanf("%s",temp->AUTHOR);
   printf("\nEnter the EDITION:");
   scanf("%s",temp->EDITION);
   temp->rlink=temp->llink=NULL;

   if(first==NULL)
   {
      first=temp;
      return first;
   }

   cur=first;
   while(cur->rlink!=NULL)
   {
      cur=cur->rlink;
   }

   cur->rlink=temp;
   temp->llink=cur;

   return first;
}
```

/*function to insert a node into DLL at the front-end*/

```
NODE insert_front(NODE first)
{
   NODE temp,cur;
   temp=getnode();
   //temp->rlink=temp->llink=NULL;
   printf("\nEnter the BOOK_ID:");
   scanf("%d",&temp->BOOK_ID);
   printf("\nEnter the BOOK_TITLE:");
   scanf("%s",temp->BOOK_TITLE);
   printf("\nEnter the AUTHOR:");
   scanf("%s",temp->AUTHOR);
   printf("\nEnter the EDITION:");
   scanf("%s",temp->EDITION);
   temp->rlink=temp->llink=NULL;

   if(first==NULL)
   {
      first=temp;
      return first;
   }

   temp->rlink=first;
   first->llink=temp;

   return temp;
}
```

/*function to count the number of nodes in a DLL*/

```
NODE count_nodes(NODE first)
{
   int count=0;
   NODE cur;
   if(first==NULL)
   {
      printf("\nList is Empty\n");
      return NULL;
   }

   cur=first;
   while(cur!=NULL)
   {
      count++;
      cur=cur->rlink;
   }

   printf("\nThe no. of nodes in the list = %d\n",count);
   return first;
}
```

/*function to delete a node from the rear-end in a DLL*/

```
NODE del_rear(NODE first)
{
   NODE cur,prev;

   if(first==NULL)
   {
      printf("\nList is Empty\n");
      return NULL;
   }

   prev=NULL;
   cur=first;
   while(cur->rlink!=NULL)
   {
      prev=cur;
      cur=cur->rlink;
   }
   printf("\n The node with the following information is Deleted:");
   printf("\nBOOK_ID:%d",cur->BOOK_ID);
   printf("\nBOOK_TITLE:%s",cur->BOOK_TITLE);
   printf("\nAUTHOR:%s",cur->AUTHOR);
   printf("\nEDITION:%s",cur->EDITION);
   free(cur);

   prev->rlink=NULL;
   return first;
}

/*function to delete a node from the front-end in a DLL*/
NODE del_front(NODE first)
{
   NODE cur;

   if(first==NULL)
   {
      printf("\nList is Empty\n");
      return NULL;
   }

   cur=first;
   first=first->rlink;
   printf("\n The node with the following information is Deleted:");
   printf("\nBOOK_ID:%d",cur->BOOK_ID);
   printf("\nBOOK_TITLE:%s",cur->BOOK_TITLE);
   printf("\nAUTHOR:%s",cur->AUTHOR);
   printf("\nEDITION:%s",cur->EDITION);
   free(cur);

   first->llink=NULL;

   return first;
}

/*function to display the node information in a DLL*/
void display(NODE first)
{
   NODE temp;
   if(first==NULL)
   {
      printf("\nDList is Empty...\n");
      return;

   }
   printf("\nThe contents of the DList are:\n");
   for(temp=first;temp!=NULL;temp=temp->rlink)
   {
      printf("BOOK_ID=%d\n",temp->BOOK_ID);
      printf("BOOK_TITLE=%s\n",temp->BOOK_TITLE);
      printf("AUTHOR=%s\n",temp->AUTHOR);
      printf("EDITION=%s\n",temp->EDITION);
      printf("\n");
   }
}
```

```
void main()
{
   NODE first=NULL;
   int choice;
   //clrscr();

   for(;;)
   {
      printf("\n*********OPERATIONS ON DLL*****************\n");
      printf("1:Insert-rear\n2:Insert-front\n3.Count the nodes in the List\n");
      printf("4:Delete-rear\n5:Delete-front\n6:Display\n7:Exit\n");
       printf("*******************************************\n");
      printf("Enter your choice:");
      scanf("%d",&choice);
      switch(choice)
      {
         case 1: first=insert_rear(first);
               break;
         case 2: first=insert_front(first);
               break;
         case 3: first=count_nodes(first);
               break;
         case 4: first=del_rear(first);
               break;
         case 5: first=del_front(first);
               break;
         case 6: display(first);
               break;
         default:printf("Invalid choice...\n");
               exit(0);
      }
   }
}
```

*****************************OUTPUT*********************************

```
*********OPERATIONS ON DLL*****************
1:Insert-rear
2:Insert-front
3.Count the nodes in the List
4:Delete-rear
5:Delete-front
6:Display
7:Exit
*******************************************
Enter your choice:1

Enter the BOOK_ID:1
Enter the BOOK_TITLE:DataStructures

Enter the AUTHOR:ReemaThareja

Enter the EDITION:2ed

*********OPERATIONS ON DLL*****************
1:Insert-rear
2:Insert-front
3.Count the nodes in the List
4:Delete-rear
5:Delete-front
6:Display
7:Exit
*******************************************
Enter your choice:6

The contents of the DList are:
BOOK_ID=1
BOOK_TITLE=DataStructures
AUTHOR=ReemaThareja
EDITION=2ed


*********OPERATIONS ON DLL*****************
1:Insert-rear
2:Insert-front
3.Count the nodes in the List
4:Delete-rear
5:Delete-front
6:Display
7:Exit
*******************************************
Enter your choice:2

Enter the BOOK_ID:2

Enter the BOOK_TITLE:ComputerOganization

Enter the AUTHOR:CarlHamachar

Enter the EDITION:5ed

*********OPERATIONS ON DLL*****************
1:Insert-rear
2:Insert-front
3.Count the nodes in the List
4:Delete-rear
5:Delete-front
6:Display
```

```
7:Exit
*******************************************
Enter your choice:6

The contents of the DList are:
BOOK_ID=2
BOOK_TITLE=ComputerOganization
AUTHOR=CarlHamachar
EDITION=5ed

BOOK_ID=1
BOOK_TITLE=DataStructures
AUTHOR=ReemaThareja
EDITION=2ed


*********OPERATIONS ON DLL*****************
1:Insert-rear
2:Insert-front
3.Count the nodes in the List
4:Delete-rear
5:Delete-front
6:Display
7:Exit
*******************************************
Enter your choice:1

Enter the BOOK_ID:3

Enter the BOOK_TITLE:WebTechnology

Enter the AUTHOR:RoberSi ebesta

Enter the EDITION:7ed

*********OPERATIONS ON DLL*****************
1:Insert-rear
2:Insert-front
3.Count the nodes in the List
4:Delete-rear
5:Delete-front
6:Display
7:Exit
*******************************************
Enter your choice:6

The contents of the DList are:
BOOK_ID=2
BOOK_TITLE=ComputerOganization
AUTHOR=CarlHamachar
```

```
EDITION=5ed

BOOK_ID=1
BOOK_TITLE=DataStructures
AUTHOR=ReemaThareja
EDITION=2ed

BOOK_ID=3
BOOK_TITLE=WebTechnology
AUTHOR=RoberSebesta
EDITION=7ed


*********OPERATIONS ON DLL*****************
1:Insert-rear
2:Insert-front
3.Count the nodes in the List
4:Delete-rear
5:Delete-front
6:Display
7:Exit
*******************************************
Enter your choice:3

The no. of nodes in the list = 3

*********OPERATIONS ON DLL*****************
1:Insert-rear
2:Insert-front
3.Count the nodes in the List
4:Delete-rear
5:Delete-front
6:Display
7:Exit
*******************************************
Enter your choice:5

 The node with the following information is Deleted:
BOOK_ID:2
BOOK_TITLE:ComputerOganization
AUTHOR:CarlHamachar
EDITION:5ed
*********OPERATIONS ON DLL*****************
1:Insert-rear
2:Insert-front
3.Count the nodes in the List
4:Delete-rear
5:Delete-front
6:Display
7:Exit
```

```
*******************************************
Enter your choice:3

The no. of nodes in the list = 2

*********OPERATIONS ON DLL*****************
1:Insert-rear
2:Insert-front
3.Count the nodes in the List
4:Delete-rear
5:Delete-front
6:Display
7:Exit
*******************************************
Enter your choice:4

 The node with the following information is Deleted:
BOOK_ID:3
BOOK_TITLE:WebTechnology
AUTHOR:RoberSebesta
EDITION:7ed
*********OPERATIONS ON DLL*****************
1:Insert-rear
2:Insert-front
3.Count the nodes in the List
4:Delete-rear
5:Delete-front
6:Display
7:Exit
*******************************************
Enter your choice:3

The no. of nodes in the list = 1

*********OPERATIONS ON DLL*****************
1:Insert-rear
2:Insert-front
3.Count the nodes in the List
4:Delete-rear
5:Delete-front
6:Display
7:Exit
*******************************************
Enter your choice:6

The contents of the DList are:
BOOK_ID=1
BOOK_TITLE=DataStructures
AUTHOR=ReemaThareja
EDITION=2ed
```

```
*********OPERATIONS ON DLL*****************
1:Insert-rear
2:Insert-front
3.Count the nodes in the List
4:Delete-rear
5:Delete-front
6:Display
7:Exit
*******************************************
Enter your choice:7
Invalid choice...
```

PROGRAM 4

Implement a menu driven Program for the following operations on Circular Linked

(i) **Create CLL of 'n' nodes of string. (insert front/rear)**
(ii) **Count the number of nodes in the CLL.**
(iii) **Delete the node at front/rear.**
(iv) **Display the contents of CLL.**

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

/*Stucture declaration for a node*/
struct node
{
        char info[10];
        struct node *link;
};

typedef struct node  *NODE;


/*function to allocate memory for a node*/
NODE getnode(void)
{
        NODE x;
        x=(NODE)malloc(sizeof(struct node));
        if(x==NULL)
        {
                printf("\n\t\tOut of memory");
                exit(0);
        }
        return x;
}

/*function to insert a node at the front end in a CLL*/
NODE insert_front(NODE last, char str[10])
{
   NODE temp;
   temp=getnode();
   strcpy(temp->info,str);
   temp->link=temp;

   if(last==NULL)
   {
      return temp;
   }
   else
      temp->link=last->link;

   last->link=temp;
   return last;
}
```

/*function to insert a node at the rear end in a CLL*/

```
NODE insert_rear(NODE last, char str[10])
{
   NODE temp;
   temp=getnode();
   strcpy(temp->info,str);
   temp->link=temp;

   if(last==NULL)
   {
      return temp;
   }
   else
      temp->link=last->link;

   last->link=temp;
   return temp;
}
```

/*function to delete the a node from the rear end in a CLL*/

```
NODE delete_front(NODE last)
{
   NODE temp;
   if(last==NULL)
   {
      printf("\nList is empty\n");
      return last;
   }
   if(last->link==last)
   {
      printf("\n \t\tThe node with info %s is deleted\n",last->info);
      free(last);
      return NULL;
   }
```

```
    temp=last->link;
    last->link=temp->link;
    printf("\n \t\tThe node with info %s is deleted\n",temp->info);
    free(temp);
    return last;
}
```

/*function to delete the a node from the front end in a CLL*/

```
NODE delete_rear(NODE last)
{
    NODE temp;
    if(last==NULL)
    {
        printf("\nList is empty\n");
        return last;
    }
    if(last->link==last)
    {
        printf("\n \t\tThe node with info %s is deleted\n",last->info);
        free(last);
        return NULL;
    }
    temp=last->link;
    while(temp->link!=last)
    {
        temp=temp->link;
    }
    temp->link=last->link;
    printf("\n \t\tThe node with info %s is deleted\n",last->info);
    free(last);
    return temp;
}
```

/*function to count the number of nodes in a CLL*/

```
void count_nodes(NODE last)
{
    int count=0;
    NODE cur;
    if(last==NULL)
    {
        printf("\n\t\tList is Empty\n");
        return;
    }
    cur=last->link;
    while(cur!=last)
    {
        count++;
        cur=cur->link;
    }
    count++;
    printf("\n\t\tThe no. of nodes in the list = %d\n",count);

}

/*function to display the contents of CLL*/
void display(NODE last)
{
    NODE cur;
    if(last==NULL)
    {
        printf("\n\t\tList is Empty\n");
        return;
    }

    cur=last->link;
    while(cur!=last)
    {
        printf("%s\n",cur->info);
        cur=cur->link;
    }
    printf("%s\n",cur->info);
}


void main()
{
    NODE last=NULL;
    int choice;
    char str[20];
     for(;;)
    {
        printf("************OPERATIONS ON CLL****************\n");
        printf("1:Insert Front\n2:Insert Rear\n3:Delete Front\n4:Delete Rear\n");
        printf("5:Count the nodes in the List\n6:Display\n7:Exit\n");
         printf("*********************************************\n");
        printf("Enter the choice:");
        scanf("%d",&choice);
        switch(choice)
        {
           case 1:
                     printf("\nEnter the string data:");
                     scanf("%s",str);
                      last=insert_front(last,str);
                     break;
           case 2:
                      printf("\nEnter the string data:");
                     scanf("%s",str);
                      last=insert_rear(last,str);
                     break;
           case 3:
                     last=delete_front(last);
                      break;
           case 4:
                     last=delete_rear(last);
                      break;
           case 5:
                     count_nodes(last);
                     break;
           case 6:
                     display(last);
                      break;
           default:
                     printf("Invalid choice...\n");
                     exit(0);
        }
     }

}
```

```
*******************************OUTPUT**********************************
***********OPERATIONS ON CLL***************
1:Insert Front
2:Insert Rear
3:Delete Front
4:Delete Rear
5:Count the nodes in the List
6:Display
7:Exit
*******************************************
Enter the choice:1

Enter the string data:a
************OPERATIONS ON CLL***************
```

```
1:Insert Front
2:Insert Rear
3:Delete Front
4:Delete Rear
5:Count the nodes in the List
6:Display
7:Exit
*******************************************
Enter the choice:1

Enter the string data:b
************OPERATIONS ON CLL****************
1:Insert Front
2:Insert Rear
3:Delete Front
4:Delete Rear
5:Count the nodes in the List
6:Display
7:Exit
*******************************************
Enter the choice:6
b
a
************OPERATIONS ON CLL****************
1:Insert Front
2:Insert Rear
3:Delete Front
4:Delete Rear
5:Count the nodes in the List
6:Display
7:Exit
*******************************************
Enter the choice:2

Enter the string data:c
************OPERATIONS ON CLL****************
1:Insert Front
2:Insert Rear
3:Delete Front
4:Delete Rear
5:Count the nodes in the List
6:Display
7:Exit
*******************************************
Enter the choice:6
b
a
c
************OPERATIONS ON CLL****************
1:Insert Front
```

```
2:Insert Rear
3:Delete Front
4:Delete Rear
5:Count the nodes in the List
6:Display
7:Exit
*******************************************
Enter the choice:2

Enter the string data:d
************OPERATIONS ON CLL****************
1:Insert Front
2:Insert Rear
3:Delete Front
4:Delete Rear
5:Count the nodes in the List
6:Display
7:Exit
*******************************************
Enter the choice:5

                The no. of nodes in the list = 4
************OPERATIONS ON CLL****************
1:Insert Front
2:Insert Rear
3:Delete Front
4:Delete Rear
5:Count the nodes in the List
6:Display
7:Exit
*******************************************
Enter the choice:3

                The node with info b is deleted
************OPERATIONS ON CLL****************
1:Insert Front
2:Insert Rear
3:Delete Front
4:Delete Rear
5:Count the nodes in the List
6:Display
7:Exit
*******************************************
Enter the choice:6
a
c
d
************OPERATIONS ON CLL****************
1:Insert Front
2:Insert Rear
```

```
3:Delete Front
4:Delete Rear
5:Count the nodes in the List
6:Display
7:Exit
*******************************************
Enter the choice:4

                    The node with info d is deleted
************OPERATIONS ON CLL***************
1:Insert Front
2:Insert Rear
3:Delete Front
4:Delete Rear
5:Count the nodes in the List
6:Display
7:Exit
*******************************************
Enter the choice:6
a
c
************OPERATIONS ON CLL***************
1:Insert Front
2:Insert Rear
3:Delete Front
4:Delete Rear
5:Count the nodes in the List
6:Display
7:Exit
*******************************************
Enter the choice:5

                    The no. of nodes in the list = 2
************OPERATIONS ON CLL***************
1:Insert Front
2:Insert Rear
3:Delete Front
4:Delete Rear
5:Count the nodes in the List
6:Display
7:Exit
*******************************************
Enter the choice:7
Invalid choice...
```

PROGRAM 5

Implement a menu driven Program for the following operations on STACK of Integers (Array Implementation of Stack with maximum size MAX)

- **(i) Push an Element on to Stack (Handle the situation of overflow)**
- **(ii) Pop an Element from Stack (Handle the situation of underflow)**
- **(iii) Display the contents of Stack**

```
#include<stdio.h>
#include<stdlib.h>
#include<conio.h>
#define MAX 5

/*function to push an element into stack*/
void push(int item,int s[],int *top)
{
   if(*top==MAX-1)
   {
      printf("stack overflow\n");
      return;
   }
   (*top)++;
   s[*top]=item;
   printf("\t\t%d is succesfully inserted\n", item);
}

/*function to pop an element from the stack*/
void pop(int s[],int *top)
{
   if(*top==-1)
   {
      printf("stack is empty\n");
      return;
   }
   printf("\t\t%d element is deleted",s[*top]);
   (*top)--;
}

/*function to display the contents of the stack*/
void display(int s[],int top)
{
   int i;
   if(top!=-1)
   {
      printf("stack contains :\n");
      for (i=0;i<=top;i++)
      printf("%d\n", s[i]);
   }
   else
      printf("stack is empty\n");
}

void main()
{
    int choice,item,s[MAX],top=-1;
    for(;;)
    {
        printf("\n*****************");
        printf("\n1.push \n  2.pop\n  3.display\n  4.exit\n");
        printf("\n*****************");
        printf("\nEnter your choice:");
        scanf("%d",&choice);
        switch(choice)
        {
            case 1:
                        printf("\nEnter an item to be inserted:");
                        scanf("%d",&item);
                         push(item,s,&top);
                         break;
            case 2:
                        pop(s,&top);
                         break;
            case 3:
                        display(s,top);
                        break;
            default:
                        printf("\ninvaild choice\n");
                        exit(0);
        }
    }
}
```

*******************************OUTPUT***********************************

```
*****************
1.push
  2.pop
  3.display
  4.exit

*****************
Enter your choice:3
stack is empty

*****************
1.push
  2.pop
  3.display
  4.exit
```

```
*****************
Enter your choice:1

Enter an item to be inserted:10
                        10 is succesfully inserted

*****************
1.push
  2.pop
  3.display
  4.exit

*****************
Enter your choice:1

Enter an item to be inserted:20
                        20 is succesfully inserted

*****************
1.push
  2.pop
  3.display
  4.exit

*****************
Enter your choice:1

Enter an item to be inserted:30
                        30 is succesfully inserted

*****************
1.push
  2.pop
  3.display
  4.exit

*****************
Enter your choice:1

Enter an item to be inserted:40
                        40 is succesfully inserted

*****************
1.push
  2.pop
  3.display
  4.exit

*****************
Enter your choice:1
Enter an item to be inserted:50
                        50 is succesfully inserted

*****************
1.push
  2.pop
  3.display
  4.exit

*****************
Enter your choice:1

Enter an item to be inserted:60
stack overflow

*****************
1.push
  2.pop
  3.display
  4.exit

*****************
Enter your choice:2
                        50 element is deleted
*****************
1.push
  2.pop
  3.display
  4.exit

*****************
Enter your choice:2
                        40 element is deleted
*****************
1.push
  2.pop
  3.display
  4.exit

*****************
Enter your choice:2
                        30 element is deleted
*****************
1.push
  2.pop
  3.display
  4.exit

*****************
```

```
Enter your choice:2
                        20 element is deleted
*****************
1.push
  2.pop
  3.display
  4.exit

*****************
Enter your choice:2
                        10 element is deleted
*****************
1.push
  2.pop
  3.display
  4.exit

*****************
Enter your choice:2
stack is empty

*****************
1.push
  2.pop
  3.display
  4.exit

*****************
Enter your choice:3
stack is empty

*****************
1.push
  2.pop
  3.display
  4.exit

*****************
Enter your choice:4

invalid choice
```

PROGRAM 6

Implement a Program to convert a given infix expression to postfix expression.

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

#define max 20
int top=-1;

/* Function that returns Input Precedence values */
int G(char symb)
{
   switch(symb)
   {
      case '+':
      case '-':return 1;

      case '*':
      case '%':
      case '/':return 3;

      case '^':
      case '$':return 6;

      case '(':return 9;

      case ')':return 0;

      default:return 7;
   }
}

/* Function that returns Stack Precedence values */
int F(char symb)
{
   switch (symb)
   {

      case '+':
      case '-':return 2;

      case '*':
      case '%':
      case '/':return 4;

      case '$':
      case '^':return 5;
```

```
        case '(':return 0;

        case '#':return -1;

        default:return 8;
    }
}
```

/* Function to convert an infix expression to postfix expression */

```
void infix_postfix(char inf[],char pos[])
{
    int i,j=0;
    char symb;
    char s[30];
    s[++top]='#';
    for(i=0;i<strlen(inf);i++)
    {
        symb=inf[i];
        while(F(s[top])>G(symb))
        {
            pos[j++]=s[top--];
        }

        if(F(s[top])!=G(symb))
             s[++top]=symb;

        else

             top--;
    }

    while(s[top]!='#')
    {
          pos[j++]=s[top--];
    }

    pos[j]='\0';

    printf("Expression in Postfix Format: ");
    for(i=0;i<strlen(inf);i++)
    printf("%c",pos[i]);
}



void main()
{
    char inf[30],pos[30];
    printf("Enter an Expression in Infix format:\n");
    scanf("%s",inf);
    infix_postfix(inf,pos);
}
```

```
********************************OUTPUT*********************************
Enter an Expression in Infix format:
a+b-c/d
Expression in Postfix Format: ab+cd/-

Enter an Expression in Infix format:
((a+b)*c-(d-e))^(f+g)
Expression in Postfix Format: ab+c*de--fg+^
```

PROGRAM 7

Implement the following using recursion:

- **(i) Tower_of_Hanoi**
- **(ii) GCD of two numbers**
- **(iii) Largest of ‘n’ numbers**

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

/* Function to find maximum amongst n elements */
int findMaxElement(int arr[],int size)
{
   int static i=0, max=-999;
   if(i<size)
   {
      if(max<arr[i])
      {
         max=arr[i];
      }
      i++;
      findMaxElement(arr,size);
   }
      return max;
}

/* Function to find product of two natural numbers */
int product(int a,int b)
{
   if(a<b)
   {
      return product(b,a);
   }
   else if(b!=0)
   {
      return (a+product(a,b-1));
   }
   else
   return 0;
}

/* Function definition for Tower_of_hanoi */
void tower(int n,char src,char aux,char dest)
{
   if(n==1)
   {
      printf("Move disk %d from peg %c to peg %c",n,src,dest);
      return;
```

```
    }
    tower((n-1),src,dest,aux);
    printf("\nMove disk %d from peg %c to peg %c\n",n,src,dest);
    tower((n-1),aux,src,dest);
}


void main()
{
    int arr[20];
    int i,n,max;
    int a,b,res;
    int num,ch;
    //clrscr();
    for(;;)
    {

        printf(" \n**********************************************\n");
        printf(" 1 : find Max Element\n");
        printf(" 2 : Multiplication of two natural numbers\n");
        printf(" 3 : Tower of Hanoi\n");
        printf(" 4 : Exit:Any other choice : \n");
        printf(" **********************************************\n");
        printf("Enter your choice\n");
        scanf("%d",&ch);
        switch(ch)
        {
            case 1:
                        printf("Enter the size of the array\n");
                        scanf("%d",&n);
                        printf("Enter %d elements of an array\n",n);
                        for(i=0;i<n;i++)
                        {
                                    scanf("%d",&arr[i]);
                        }
                        max=findMaxElement(arr,n);
                        printf("Maximum element in the array is %d\n",max);
                        break;
            case 2:
                        printf("Enter two numbers to find their product:\n");
                        scanf("%d%d",&a,&b);
                        res=product(a,b);
                        printf("Product of %d and %d is %d\n",a,b,res);
                        break;
            case 3:
                        printf("Enter the number of disks :\n");
                        scanf("%d",&num);
                        printf("The sequence of moves involved in the Tower of Hanoi are:\n");
                        tower(num,'S','T','D');
                        break;
            default:
                        exit(0);
        }
    }
}**********************************OUTPUT******************************
*******************************************
 1 : find Max Element
 2 : Multiplication of two natural numbers
 3 : Tower of Hanoi
 4 : Exit:Any other choice :
 *******************************************
Enter your choice
1
Enter the size of the array
6
Enter 6 elements of an array
100
50
200
250
75
60
Maximum element in the array is 250
 *******************************************
 1 : find Max Element
 2 : Multiplication of two natural numbers
 3 : Tower of Hanoi
 4 : Exit:Any other choice :
 *******************************************
Enter your choice
2
Enter two numbers to find their product:
7 8
Product of  7 and 8 is 56
 *******************************************
 1 : find Max Element
 2 : Multiplication of two natural numbers
 3 : Tower of Hanoi
 4 : Exit:Any other choice :
 *******************************************
Enter your choice
3
Enter the number of disks :
3
The sequence of moves involved in the Tower of Hanoi are:
Move disk 1 from peg S to peg D
Move disk 2 from peg S to peg T
Move disk 1 from peg D to peg T
Move disk 3 from peg S to peg D
Move disk 1 from peg T to peg S
Move disk 2 from peg T to peg D
Move disk 1 from peg S to peg D
 *****************************************
 1 : find Max Element
 2 : Multiplication of two natural numbers
 3 : Tower of Hanoi
 4 : Exit:Any other choice :
 *****************************************
Enter your choice
4
```

PROGRAM 8

Implement a menu driven Program for the following operations on QUEUES of Strings using Linked list

- **(i) Insert an Element into Queue**
- **(ii) Delete an Element from Queue**
- **(iii) Display the contents of Queue**

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

#define QS 5

/*Structure declaration for a node*/
struct node
{
	char info[10];
	struct node *link;
};

typedef struct node *NODE;


/*function to allocate memory for a node*/
NODE getnode(void)
{
	NODE x;
	x=(NODE)malloc(sizeof(struct node));
	if(x==NULL)
	{
		printf("\nOut of memory");
		exit(0);
	}
	return x;
}


/*function to insert a node at the rear end of SLL as a queue*/
NODE insert_rear(char item[10],NODE first,int *r)
{
	NODE temp,cur;
	if(*r==QS-1)
	{
	   printf("\n Queue is Full! insertion not possible\n");
	   return first;
	}
	temp=getnode();
```

```
          strcpy(temp->info,item);
          temp->link=NULL;

          if(*r==-1)
          {
             *r=*r+1;
             printf("r=%d",*r);
             first=temp;
                   return first;
          }
          else
   {
      *r=*r+1;
             cur=first;
             while(cur->link!=NULL)
             {
                      cur=cur->link;
             }
   }
          cur->link=temp;
          return first;
}
```

/*function to delete a node from the front end of SLL as a queue*/

```
NODE delete_front(NODE first, int *f, int *r)
{
   NODE temp;
   if(*f>*r)
   {
      printf("\n Queue is Empty! deletion not possible\n");
      *f=0;
      *r=-1;
            return first;
   }
   temp=first;
   first=first->link;
   printf("The node with info [ %s ] is deleted from the front end",temp->info);
   (*f)++;
   free(temp);

   return first;
}
```

/*function to display contents of the SLL as a queue*/

```
void display(NODE first, int f, int r)
{
          NODE temp;
          int i;
          if(f>r)
          {
```

```
		printf("\nList is empty");
		return;
	}
	printf("\nThe contents of the list:");
	temp=first;
	for(i=f;i<=r;i++)
	{
		printf("%s\n",temp->info);
		temp=temp->link;
	}
	printf("\n");
}

int main()
{
	NODE first=NULL;
	int choice,f=0,r=-1;
	char item[10];
	//clrscr();
	do
	{
		printf("\n************************");
		printf("\n 1: Queue_Insert\n 2: Queue_delete\n 3: Queue_Display\n 4:Exit\n");
		printf("\n************************");
		printf("\nEnter the choice:");
		scanf("%d",&choice);
		switch(choice)
		{
			case 1:
				printf("\nEnter the item to be inserted into the List(string data):");
				scanf("%s",item);
				first=insert_rear(item,first,&r);
				 break;

			case 2:
				first=delete_front(first,&f,&r);
				break;

			case 3:
				display(first,f,r);
				break;

		}
	}while(choice!=4);
	getch();
	return 0;
}
```

```
**********************************OUTPUT*******************************

************************
```

```
 1: Queue_Insert
 2: Queue_delete
 3: Queue_Display
 4: Exit
************************
Enter the choice:1

Enter the item to be inserted into the List(string data):aaaa

************************
 1: Queue_Insert
 2: Queue_delete
 3: Queue_Display
 4: Exit
************************
Enter the choice:1

Enter the item to be inserted into the List(string data):bbbb

************************
 1: Queue_Insert
 2: Queue_delete
 3: Queue_Display
 4: Exit
************************
Enter the choice:1

Enter the item to be inserted into the List(string data):cccc

************************
 1: Queue_Insert
 2: Queue_delete
 3: Queue_Display
 4: Exit
************************
Enter the choice:1

Enter the item to be inserted into the List(string data):dddd

************************
 1: Queue_Insert
 2: Queue_delete
 3: Queue_Display
 4: Exit
************************
Enter the choice:1

Enter the item to be inserted into the List(string data):eeee

************************
```

```
 1: Queue_Insert
 2: Queue_delete
 3: Queue_Display
 4: Exit
************************
Enter the choice:1

Enter the item to be inserted into the List(string data):ffff

 Queue is Full! insertion not possible
************************
 1: Queue_Insert
 2: Queue_delete
 3: Queue_Display
 4: Exit
************************
Enter the choice:3

The contents of the list:
aaaa
bbbb
cccc
dddd
eeee

************************
 1: Queue_Insert
 2: Queue_delete
 3: Queue_Display
 4: Exit
************************
Enter the choice:2
The node with info [ aaaa ] is deleted from the front end
************************
 1: Queue_Insert
 2: Queue_delete
 3: Queue_Display
 4: Exit
************************
Enter the choice:2
The node with info [ bbbb ] is deleted from the front end
************************
 1: Queue_Insert
 2: Queue_delete
 3: Queue_Display
 4: Exit
************************
Enter the choice:2
The node with info [ cccc ] is deleted from the front end
************************
```

```
 1: Queue_Insert
 2: Queue_delete
 3: Queue_Display
 4: Exit
************************
Enter the choice:1

Enter the item to be inserted into the List(string data):tttt

 Queue is Full! insertion not possible

************************
 1: Queue_Insert
 2: Queue_delete
 3: Queue_Display
 4: Exit
************************
Enter the choice:3

The contents of the list:
dddd
eeee

************************
 1: Queue_Insert
 2: Queue_delete
 3: Queue_Display
 4: Exit
************************
Enter the choice:2
The node with info [ dddd ] is deleted from the front end
************************
 1: Queue_Insert
 2: Queue_delete
 3: Queue_Display
 4: Exit

************************
Enter the choice:2
The node with info [ eeee ] is deleted from the front end
************************
 1: Queue_Insert
 2: Queue_delete
 3: Queue_Display
 4: Exit
************************
Enter the choice:2

 Queue is Empty! deletion not possible
```

```
************************
 1: Queue_Insert
 2: Queue_delete
 3: Queue_Display
 4: Exit
************************
Enter the choice:3

List is empty
************************
 1: Queue_Insert
 2: Queue_delete
 3: Queue_Display
 4: Exit
************************
Enter the choice:1

Enter the item to be inserted into the List(string data):zzzz

************************
 1: Queue_Insert
 2: Queue_delete
 3: Queue_Display
 4: Exit
************************
Enter the choice:4
```

PROGRAM 9

Implement a menu driven program to perform the following operations on priority queue using linked list.

(i) **Insert a node based on priority.**

(ii) **Delete a node from the queue**

(iii) **Display the contents of the queue**

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

/*Stucture declaration*/
struct node
{
int data;
int priority;
struct node *link;
};

typedef struct node *NODE;

/*function to allocate memory for a node*/
NODE getnode(void)
{
	NODE x;
	x=(NODE)malloc(sizeof(struct node));
	if(x==NULL)
	{
		printf("\nOut of memory");
		exit(0);
	}
	return x;
}



/*function to insert a node based on priority*/
NODE insert_priority(NODE first)
{
   int val, pri;
   NODE temp,cur,prev;
   printf("\n Enter the value:");
   scanf("%d", &val);
   printf("\n Enter the priority:");
   scanf("%d", &pri);
   temp=getnode();
   temp->data=val;
```

```
    temp->priority= pri;
    temp->link=NULL;
    if(first==NULL || pri < first->priority )
            {
                temp->link = first;
                first = temp;
                return first;
            }
            prev=NULL;
            cur = first;
            //next= cur->link;
            while(cur!=NULL && cur->priority<=pri)
            {
                prev=cur;
                cur = cur->link;
            }
            if(cur==NULL)
            {
                prev->link=temp;
                return first;
            }
            temp->link = cur;
            prev->link = temp;

            return first;
}
```

/*function to delete a node from the front end of the priority queue*/

```
NODE delete(NODE first)
{
    NODE temp;
    if(first == NULL)
            {
                printf("\n QUEUE UNDERFLOW" );
                return NULL;
            }
            temp = first;
            printf("\n Deleted item is: %d", temp->data);
            first = first->link;
            free(temp);
    return first;
}
```

/*function to display the contents of priority queue*/

```
void display(NODE first)
{
    NODE temp;
    temp = first;
```

```
   if(first == NULL)
         {
            printf("\nQUEUE IS EMPTY" );
            return;
         }
         printf("\n THE CONTENTS OF PRIORITY QUEUE IS : " );
         while(temp != NULL)
         {
                  printf( "\n%d [priority=%d]", temp->data, temp->priority );
                  temp=temp->link;
         }

}

void main()
{
   NODE first=NULL;
   int option;
   //clrscr();
   do
   {
      printf("\n *****MAIN MENU*****");
      printf("\n 1. INSERT");
      printf("\n 2. DELETE");
      printf("\n 3. DISPLAY");
      printf("\n 4. EXIT");
      printf("\n Enter your option : ");
      scanf( "%d", &option);
      switch(option)
      {
         case 1:
                  first=insert(first);
                  break;
         case 2:
                     first = delete(first);
                     break;
         case 3:
                  display(first);
                     break;
      }
   }while(option!=4);
}
```

**********************************OUTPUT********************************

```
 *****MAIN MENU*****
 1. INSERT
 2. DELETE
 3. DISPLAY
 4. EXIT
 Enter your option : 2

 QUEUE UNDERFLOW
 *****MAIN MENU*****
 1. INSERT
 2. DELETE
 3. DISPLAY
 4. EXIT
 Enter your option : 1

 Enter the value:10

 Enter the priority:4

 *****MAIN MENU*****
 1. INSERT
 2. DELETE
 3. DISPLAY
 4. EXIT
 Enter your option : 3

 THE CONTENTS OF PRIORITY QUEUE IS :
10 [priority=4]
 *****MAIN MENU*****
 1. INSERT
 2. DELETE
 3. DISPLAY
 4. EXIT
 Enter your option : 1

 Enter the value:20

 Enter the priority:2

 *****MAIN MENU*****
 1. INSERT
 2. DELETE
 3. DISPLAY
 4. EXIT
 Enter your option : 3

 THE CONTENTS OF PRIORITY QUEUE IS :
20 [priority=2]
10 [priority=4]
 *****MAIN MENU*****
 1. INSERT
 2. DELETE
 3. DISPLAY
 4. EXIT
 Enter your option : 1

 Enter the value:30

 Enter the priority:3

 *****MAIN MENU*****
 1. INSERT
 2. DELETE
 3. DISPLAY
 4. EXIT
 Enter your option : 3

 THE CONTENTS OF PRIORITY QUEUE IS :
20 [priority=2]
30 [priority=3]
10 [priority=4]
 *****MAIN MENU*****
 1. INSERT
 2. DELETE
 3. DISPLAY
 4. EXIT
 Enter your option : 1

 Enter the value:40

 Enter the priority:2

 *****MAIN MENU*****
 1. INSERT
 2. DELETE
 3. DISPLAY
 4. EXIT
 Enter your option : 3

 THE CONTENTS OF PRIORITY QUEUE IS :
20 [priority=2]
40 [priority=2]
30 [priority=3]
10 [priority=4]
 *****MAIN MENU*****
 1. INSERT
 2. DELETE
 3. DISPLAY
 4. EXIT
 Enter your option : 1

 Enter the value:60

 Enter the priority:3
 *****MAIN MENU*****
 1. INSERT
 2. DELETE
 3. DISPLAY
 4. EXIT
 Enter your option : 3

 THE CONTENTS OF PRIORITY QUEUE IS :
20 [priority=2]
40 [priority=2]
30 [priority=3]
60 [priority=3]
10 [priority=4]
 *****MAIN MENU*****
 1. INSERT
 2. DELETE
 3. DISPLAY
 4. EXIT
 Enter your option : 1

 Enter the value:50

 Enter the priority:4

 *****MAIN MENU*****
 1. INSERT
 2. DELETE
 3. DISPLAY
 4. EXIT
 Enter your option : 3

 THE CONTENTS OF PRIORITY QUEUE IS :
20 [priority=2]
40 [priority=2]
30 [priority=3]
60 [priority=3]
10 [priority=4]
50 [priority=4]
 *****MAIN MENU*****
 1. INSERT
 2. DELETE
 3. DISPLAY
 4. EXIT
 Enter your option : 1

 Enter the value:100

 Enter the priority:1
 *****MAIN MENU*****
 1. INSERT
 2. DELETE
 3. DISPLAY
 4. EXIT
 Enter your option : 3

 THE CONTENTS OF PRIORITY QUEUE IS :
100 [priority=1]
20 [priority=2]
40 [priority=2]
30 [priority=3]
60 [priority=3]
10 [priority=4]
50 [priority=4]
 *****MAIN MENU*****
 1. INSERT
 2. DELETE
 3. DISPLAY
 4. EXIT
 Enter your option : 2

 Deleted item is: 100
 *****MAIN MENU*****
 1. INSERT
 2. DELETE
 3. DISPLAY
 4. EXIT
 Enter your option : 2

 Deleted item is: 20
 *****MAIN MENU*****
 1. INSERT
 2. DELETE
 3. DISPLAY
 4. EXIT
 Enter your option : 3

 THE CONTENTS OF PRIORITY QUEUE IS :
40 [priority=2]
30 [priority=3]
60 [priority=3]
10 [priority=4]
50 [priority=4]
 *****MAIN MENU*****
 1. INSERT
 2. DELETE
 3. DISPLAY
 4. EXIT
 Enter your option : 2
 Deleted item is: 40
 *****MAIN MENU*****
 1. INSERT
 2. DELETE
 3. DISPLAY
 4. EXIT
 Enter your option : 2

 Deleted item is: 30
 *****MAIN MENU*****
 1. INSERT
 2. DELETE
 3. DISPLAY
 4. EXIT
 Enter your option : 2

 Deleted item is: 60
 *****MAIN MENU*****
 1. INSERT
 2. DELETE
 3. DISPLAY
 4. EXIT
 Enter your option : 3

 THE CONTENTS OF PRIORITY QUEUE IS :
10 [priority=4]
50 [priority=4]
 *****MAIN MENU*****
 1. INSERT
 2. DELETE
 3. DISPLAY
 4. EXIT
 Enter your option : 4
```

PROGRAM 10

Implement a menu driven Program for the following operations on Binary Search Tree (BST) of Integers

(i) **Create a BST of N Integers**
(ii) **Traverse the BST in Inorder, Preorder and Postorder**

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

int flag;

/*structure declaration for a node*/
struct node
{
   int info;
   struct node *llink,*rlink;
};
typedef struct node *NODE;


/*function to allocate memory for a node*/
NODE getnode()
{
   NODE x;
   x=(NODE)malloc(sizeof(struct node));
   if(x==NULL)
   {
      printf("out of memory\n");
      exit(0);
   }
return x;
}


/*function to insert a node into BST*/
NODE insert(NODE root,int item)
{
   NODE temp,cur,prev;
   temp=getnode();
   temp->info=item;
   temp->llink=temp->rlink=NULL;
   if(root==NULL)
      return temp;
   prev =NULL;
   cur=root;
   while(cur!=NULL)
   {
      prev=cur;
      if(item==cur->info)
       {
         printf("duplicate items are not allowed\n");
         free(temp);
         return root;
      }
      if(item<cur->info)
         cur=cur->llink;
      else
         cur=cur->rlink;
   }
   if(item<prev->info)
      prev->llink=temp;
   else
      prev->rlink=temp;
   return root;
}


/*function for inorder traversal*/
void in_order(NODE root)
{
   if(root!=NULL)
   {
      in_order(root->llink);
      printf("%d\t",root->info);
      in_order(root->rlink);
   }
}


/*function for preorder traversal*/
void pre_order(NODE root)
{
   if(root!=NULL)
   {
       printf("%d\t",root->info);
       pre_order(root->llink);
      pre_order(root->rlink);
   }
}


/*function for postorder traversal*/
void post_order(NODE root)
{
   if(root!=NULL)
   {
      post_order(root->llink);
```

```
        post_order(root->rlink);
        printf("%d\t",root->info);
    }
}


void main()
{
    NODE root=NULL;
    int ch,item;
    //clrscr();
    for(;;)
    {
        printf("\n1.BST Insertion\n 2.preorder\n 3.inorder\n 4.postorder\n 5.exit\n");
        printf("\nEnter your choice:");
        scanf("%d",&ch);
        switch(ch)
        {
            case 1:
                        printf("\nEnter the item to be inserted:");
                        scanf("%d",&item);
                         root=insert(root,item);
                         break;

            case 2:
                         printf("pre traversal is\n");
                        pre_order(root);
                        break;
            case 3:
                        printf("in traversal is\n");
                         in_order(root);
                        break;
            case 4:
                        printf("post traversal is\n" );
                        post_order(root);
                         break;
            default:
                        exit(0);
        }
    }
}
```

**********************************OUTPUT********************************

```
1.BST Insertion
 2.preorder
 3.inorder
 4.postorder
 5.exit
```

```
Enter your choice:1

Enter the item to be inserted:100

1.BST Insertion
 2.preorder
 3.inorder
 4.postorder
 5.exit

Enter your choice:1

Enter the item to be inserted:80

1.BST Insertion
 2.preorder
 3.inorder
 4.postorder
 5.exit

Enter your choice:1

Enter the item to be inserted:250

1.BST Insertion
 2.preorder
 3.inorder
 4.postorder
 5.exit

Enter your choice:1

Enter the item to be inserted:70

1.BST Insertion
 2.preorder
 3.inorder
 4.postorder
 5.exit

Enter your choice:1

Enter the item to be inserted:90

1.BST Insertion
 2.preorder
 3.inorder
 4.postorder
 5.exit
```

```
Enter your choice:1

Enter the item to be inserted:200

1.BST Insertion
 2.preorder
 3.inorder
 4.postorder
 5.exit

Enter your choice:1

Enter the item to be inserted:270

1.BST Insertion
 2.preorder
 3.inorder
 4.postorder
 5.exit

Enter your choice:2
pre traversal is
100     80      70      90      250     200     270
1.BST Insertion
 2.preorder
 3.inorder
 4.postorder
 5.exit

Enter your choice:3
in traversal is
70      80      90      100     200     250     270
1.BST Insertion
 2.preorder
 3.inorder
 4.postorder
 5.exit

Enter your choice:4
post traversal is
70      90      80      200     270     250     100
1.BST Insertion
 2.preorder
 3.inorder
 4.postorder
 5.exit

Enter your choice:5
```

**********END**********