1. Write a program to implement push, pop, peep, search & display in stack through array representation.

```
#include<stdio.h>
#include<stdlib.h>
#define max_size 5
int stack[max_size],top=-1;

void push()
{int item;
 if(top==(max_size-1))
 {printf("\nStack Overflow"); }
 else
 {
 printf("Enter the element to be inserted:\t");
 scanf("%d",&item);
 top=top+1;
 stack[top]=item;
 }
}

void pop()
{int item;
 if(top==-1)
 {printf("Stack Underflow"); }
 else
 {
 item=stack[top];
 top=top-1;
 printf("\nThe poped element: %d\t",item);
 }
}

void peep()
{if(top==-1)
 {printf("\nStack is empty"); }
 else
 {printf("The topmost element of the stack is %d",stack[top]); }
}

void search()
{
int i,ele;
if(top==-1)
{printf("\n Stack is empty"); }
else
{
printf("\nEnter the element to be searched :");
scanf("%d",&ele);
for(i=top;i>=0;i--)
{if (i==ele)
 {printf("Number found at the location = %d",i); }
 else
 {printf("Number not found"); }
  }
 }
}

void display()
{int i;
 if(top==-1)
 {printf("\nStack is Empty"); }
 else
```

```
 {
 printf("\nThe stack elements are:\n" );
 for(i=top;i>=0;i--)
 {printf("%d\n",stack[i]); }
 }
}

void main()
{int ch;
 do
 {
 printf("1.Push\n 2.Pop\n 3.Peep\n 4.Search\n 5.Display\n 6.Exit\n");
 printf("\nEnter your choice:\t");
 scanf("%d",&ch);
 switch(ch)
 {case 1: push();
                break;
  case 2: pop();
                break;
  case 3: peep();
                break;
  case 4: search();
                break;
  case 5: display();
                break;
  case 6: exit(0);
                break;
  default: printf("\nInvalid choice\n");
                 break;
  }
 }while(ch!='6');
getch();
}
```

2. Write a program to implement insert, delete, search & display in queue through array representation.

```
#include<stdio.h>
#include<conio.h>
#define MAX 50
int queue_array[MAX];
int rear=-1;
int front=-1;

void insert()
{int add_item;
 if(rear==MAX-1)
 printf("Queue Overflow \n");
 else
 {
 if(front==-1)
 front=0;
 printf("Inset the element in queue : ");
 scanf("%d",&add_item);
 rear=rear+1;
 queue_array[rear]=add_item;
 }
}
```

```
void delete()
{if((front==-1)||(front>rear))
 {printf("Queue Underflow \n");
  return;}
 else
 {printf("Element deleted from queue is :%d\n",queue_array[front]);
  front=front+1;
  }
}

void search()
{
int i,ele;
if(front==-1)
{printf("\n Queue is empty"); }
else
{
printf("\nEnter the elements to be searched :");
scanf("%d",&ele);
for(i=front;i<=rear;i++)
{if (i==ele)
 {printf("Number found at the location = %d",i); }
 else
 {printf("Number not found"); }
  }
 }
}

void display()
{int i;
 if(front==-1)
 printf("Queue is empty \n");
 else
 {
 printf("Queue is : \n");
 for(i=front;i<=rear;i++)
 printf("%d ",queue_array[i]);
 printf("\n");
 }
}

void main()
{int ch;
 while (1)
 {
 printf("1.Insert element to queue \n");
 printf("2.Delete element from queue \n");
 printf("3.Display all elements of queue \n");
 printf("4.Quit \n");
 printf("Enter your choice : ");
 scanf("%d",&ch);
 switch (ch)
 {case 1: insert();
                 break;
  case 2: delete();
                 break;
  case 3: search();
                 break;
  case 4: display();
                 break;
  case 5: exit(1);
  default: printf("Wrong choice \n");
  }
```

```
 }
getch();
}
```

3. Write a program to implement insert, delete, search & display in circular queue through array representation.

```
#include<stdio.h>
#include<conio.h>
#define max 3
int q[10],front=0,rear=-1;

void insert()
{
int x;
if((front==0&&rear==max-1)||(front>0&&rear==front-1))
printf("Queue is overflow\n");
else
{printf("Enter element to be insert:");
 scanf("%d",&x);
 if(rear==max-1&&front>0)
 {rear=0;
  q[rear]=x;
  }
 else
 {if((front==0&&rear==-1)||(rear!=front-1))
  q[++rear]=x;
  }
 }
}

void delete()
{
int a;
if((front==0)&&(rear==-1))
{printf("Queue is underflow\n");
 getch();
 exit();
 }
if(front==rear)
{a=q[front];
 rear=-1;
 front=0;
 }
else if(front==max-1)
{a=q[front];
 front=0;
 }
else
 a=q[front++];
printf("Deleted element is:%d\n",a);
}

void search()
{
int j,i,ele;
printf("\nEnter the elements to be searched :");
scanf("%d",&ele);
if(front==0&&rear==-1)
```

```
{printf("\n Queue is empty"); }
else if(front>rear)
{for(i=0;i<=rear;i++)
  {if (i==ele)
   printf("Number found at the location = %d",i);}
 for(j=front;j<=max-1;j++)
  {if (j==ele)
   printf("Number found at the location = %d",j); }
  }
else
{for(i=front;i<=rear;i++)
 {if (i==ele)
  printf("Number found at the location = %d",i); }
  }
printf("\n");
}

void display()
{
int i,j;
if(front==0&&rear==-1)
{printf("Queue is underflow\n");
 getch();
 exit(); }
if(front>rear)
{for(i=0;i<=rear;i++)
 printf("\t%d",q[i]);
 for(j=front;j<=max-1;j++)
 printf("\t%d",q[j]);
 printf("\nrear is at %d\n",q[rear]);
 printf("\nfront is at %d\n",q[front]);
 }
else
{for(i=front;i<=rear;i++)
 {printf("\t%d",q[i]);
  }
 printf("\nrear is at %d\n",q[rear]);
 printf("\nfront is at %d\n",q[front]);
 }
printf("\n");
}

void main()
{
int ch;
clrscr();
printf("1.Insert\n2.Delete\n3.Search\n4.Display\n5.Exit\n");
while(1)
{printf("Enter your choice:");
 scanf("%d",&ch);
 switch(ch)
 {case 1: insert();
                break;
  case 2: delete();
                break;
  case 3: search();
                break;
  case 4: display();
                break;
  case 5: exit(1);
  default: printf("Invalid option\n");
  }
 }
```

```
getch();
}
```

4. Write a program to implement push, pop, peep, search & display in stack through linked list representation.

```
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
struct stack
{int info;
 struct stack *next;
}*n;
typedef struct stack stack;
stack *push(stack *top,int item)
{
n=(stack*)malloc(sizeof(stack));
if(n==NULL)
printf("overflow");
else
{n->info=item;
 n->next=top;
 top=n;
 }
return top;
}

stack *pop(stack *top)
{
int del;
stack *tmp;
if(top==NULL)
printf("underflow");
else
{del=top->info;
 tmp=top;
 top=top->next;
 free(tmp);
 printf("deleted item is %d",del);
 }
return top;
}

int peep(stack *top)
{
if(top==NULL)
printf("empty stack");
else
return top->info;
}

void search(stack *top)
{
stack *ptr=top;
int ele;
if(top==NULL)
{printf("\n Stack is empty"); }
else
{
```

```
printf("\nEnter the elements to be searched :");
scanf("%d",&ele);
while(ptr!=NULL)
{if (ptr->info==ele)
 {printf("Number found at the location = %d",ptr); }
  ptr=ptr->next;
 else
 {printf("Number not found"); }
  }
 }
}

void display(stack *top)
{
stack *ptr=top;
if(top==NULL)
printf("empty list");
else
{while(ptr!=NULL)
 {
 printf("%d\t",ptr->info);
 ptr=ptr->next;
  }
 }
}

void main()
{
stack *top=NULL;
int ch,item;
while(1)
{printf("\n**********************************");
 printf("\n1. Push");
 printf("\n2. Pop");
 printf("\n3. Peep");
 printf("\n4. Search");
 printf("\n5. Display");
 printf("\n6. Exit");
 printf("\n Enter your choise:");
 scanf("%d",&ch);
 switch(ch)
 {case 1: printf("Enter information:");
                scanf("%d",&item);
     top=push(top,item);
     break;
  case 2: top=pop(top);
    break;
  case 3: printf("Peep element is %d",peep(top));
    break;
  case 4: search();
                break;
  case 5: display(top);
                break;
  case 6: exit(1);
  }
 }
getch();
}
```

5. Write a program to implement insert, delete, search & display in queue through linked list representation.

```
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
struct queue
{int data;
 struct queue *next;
}*n,*front,*rear;

void insert(int item)
{
n=(struct queue*)malloc(sizeof(struct queue));
if(n==NULL)
printf("\already full");
else
{n->data=item;
 if(front==NULL)
 front=rear=n;
 else
 {rear->next=n;
  rear=n;
  }
 rear->next=NULL;
 }
}

void delete()
{
int del;
struct queue *tmp;
if(front==NULL)
printf("empty queue");
else
{del=front->data;
 tmp=front;
 if(front==rear)
 front=rear=NULL;
 else
 front=front->next;
 printf("deleted item is %d",del);
 free(tmp);
 }
}

void search()
{
struct queue *ptr;
int ele;
if(front==NULL)
{printf("\n Queue is empty"); }
else
{
printf("\nEnter the elements to be searched :");
scanf("%d",&ele);
for(ptr=front;ptr!=NULL;ptr=ptr->next)
{if (ptr->data==ele)
 {printf("Number found at the location = %d",ptr); }
 else
 {printf("Number not found"); }
  }
 }
```

```
}

void display()
{
struct queue *ptr;
if(front==NULL)
printf("nothing to display");
else
{for(ptr=front;ptr!=NULL;ptr=ptr->next)
printf("%d\t",ptr->data);}
}

void main()
{
front=rear=NULL;
int ch,item;
while(1)
{printf("\n1. Insert element");
 printf("\n2. Delete element");
 printf("\n3. Search");
 printf("\n4. Display");
 printf("\n5. Exit");
 printf("\n Choose operation:");
 scanf("%d",&ch);
 switch(ch)
 {case 1: printf("enter element:");
    scanf("%d",&item);
    insert(item);
    break;
  case 2: delete();
    break;
  case 3: search();
                break;
  case 4: display();
    break;
  case 5: exit(1);
  }
 }
getch();
}
```

6. Write a program to implement insert, delete, search & display in circular queue through linked list representation.

```
#include<stdio.h>
#include<conio.h>
#define Maxque 10
struct Queue
{int Item[Maxque];
 int front,rear;
 }q;

void Insert(struct Queue *pq,int ele)
{
if((pq->front==0&&pq->rear==Maxque-1)||(pq->front==pq->rear+1))
printf("Overflow");
else
{if(pq->front==-1)
 pq->front=pq->rear=0;
```

```
 else if(pq->rear==Maxque-1)
 pq->rear=0;
 else
 pq->rear=pq->rear+1;
 pq->Item[pq->rear]==ele;
 }
}

void Delete(struct Queue *pq)
{
int del;
if(pq->front==-1)
printf("Underflow");
else
{del=pq->Item[pq->front];
 if(pq->front==pq->rear)
 pq->front=pq->rear=-1;
 else if(pq->front==Maxque-1)
 pq->front=0;
 else
 pq->front=pq->front+1;
 printf("Deleted element is %d",del);
 }
}

void Search(struct Queue *pq,int ele)
{
int i;
if(pq->front==-1)
{printf("\n Queue is empty"); }
else if(pq->front>pq->rear)
{for(i=pq->front;i<Maxque;i++)
  {if (i==ele)
   printf("Element found at the location = %d",i);}
 for(i=0;i<=pq->rear;i++)
  {if (i==ele)
   printf("Element found at the location = %d",i); }
  }
else
{for(i=pq->front;i<=pq->rear;i++)
 {if (i==ele)
  printf("Element found at the location = %d",i); }
  }
printf("\n");
}

void Display(struct Queue *pq)
{
int i;
if(pq->front==-1)
printf("Nothing have to display");
else
{if(pq->front>pq->rear)
 {for(i=pq->front;i<Maxque;i++)
  printf("\t %d",pq->Item[i]);
  for(i=0;i<=pq->rear;i++)
  printf("\t %d",pq->Item[i]);
   }
 else
 {for(i=pq->front;i<=pq->rear;i++)
  printf("\t %d",pq->Item[i]);
   }
 }
```

```
}

void main()
{
int ch;
clrscr();
q.front=-1;
q.rear=-1;
printf("\nCircular Queue operations using linked list\n");
printf("1. Insert\n2. Delete\n3. Search\n4. Display\n5. Exit\n");
while(1)
{printf("Enter your choice:");
 scanf("%d",&ch);
 switch(ch)
 {case 1: Insert(&q,4);
                Insert(&q,7);
                Insert(&q,3);
                break;
  case 2: Delete(&q);
                break;
  case 3: printf("\nEnter the element to be searched :");
                scanf("%d",&ele);
                Search(&q,ele);
                break;
  case 4: Display(&q);
                break;
  case 5: exit(1);
  default: printf("Invalid option\n");
  }
 }
getch();
}
```

7. Write a function to reverse a string using stack.

```
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
#include<string.h>
struct stack
{
 int info;
 struct stack *next;
}*n,*top=NULL,*tmp;
typedef struct stack stack;
void push(char item)
{
 n=(stack*)malloc(sizeof(stack));
 if(n==NULL)
 printf("overflow");
 else
 {
  n->info=item;
  n->next=top;
  top=n;
 }
}
char pop()
{
```

```
 char del;
 if(top==NULL)
 printf("underflow");
 else
 {
  del=top->info;
  tmp=top;
  top=top->next;
  free(tmp);
 }
 return del;
}
void main()
{
 char str[20];
 int i;
 printf("enter a string...\n");
 gets(str);
 for(i=0;i<strlen(str);i++)
 push(str[i]);
 for(i=0;i<strlen(str);i++)
 str[i]=pop();
 printf("reverse of string is...\n");
 puts(str);
 getch();
}
```

8. Write a recursive function to solve Tower of Hanoi.

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

void TOH(int n,char Beg,char Aux,char End)
{
 if(n==1)
 { printf("%c->%c\n",Beg,End);
   return;
}
 else
 {
  TOH(n-1,Beg,End,Aux);
  printf("%c->%c\n",Beg,End);
  TOH(n-1,Aux,Beg,End);
}
}
void main()
{
 int m;
    char A,B,C;
 printf("Enter number of elements");
 scanf("%d",&m);
 TOH(m,'A','B','C');
 getch();
}
```

9. Write a program to implement queue through doubly linked list.

```
#include<stdio.h>
#include<conio.h>
#include<malloc.h>
struct queue
{
int data;
struct queue *prev;
struct queue *next;
};
typedef struct queue *Queue;
Queue first=NULL,temp,last;

void queue_in(int data)
{
temp=(struct queue*)malloc(sizeof(struct queue));
temp->data=data;
temp->next=NULL;
if(first==NULL)
{first=last=temp;
 first->prev=NULL;
 return;
 }
else
{last->next=temp;
 temp->prev=last;
 last=temp;
 }
}

int queue_out()
{
int data;
if(first==NULL)
{printf("\n\n Queue is empty\n\n");
 return 0;
 }
if(first==last)
{data=first->data;
 free(first);
 first=NULL;
 return data;
 }
temp=first;
data=temp->data;
first=first->next;
free(temp);
return data;
}

void display()
{
if(first==NULL)
{printf("\n\n Queue is empty\n\n");
 return;
 }
printf("Queue data's \n");
temp=first;
while(temp != NULL)
{printf("%d\n",temp->data);
 temp=temp->next;
```

```
 }
}

void main()
{
int data,select;
clrscr();
while(1)
{printf(" 1:Queue_in\n 2:Queue_out\n 3:Display all data\n 4:Exit\n\t");
 scanf("%d",&select);
 switch(select)
 {case 1:
  printf("Enter the data:");
  scanf("%d",&data);
  queue_in(data);
  break;

  case 2: data=queue_out();
  if(data==0)
  break;
  printf("\n Queue out data:%d\n\n",data);
  break;

  case 3:
  display();
   break;

   case 4:
   break;
  }}
getch();
}
```

10. Write a program to convert infix expression into postfix expression.

```
#include<stdio.h>
#include<stdlib.h>
#include<conio.h>
#include<string.h>
char infix[20],postfix[20];
struct stack
{char info;
 struct stack *next;
}*s,*top=NULL;
typedef struct stack stk;
void push(char symbol)
{
s=(stk*)malloc(sizeof(stk));
 s->info=symbol;
 s->next=top;
 top=s;
}

char pop()
{
char poped;
 stk *tmp;
 poped=top->info;
```

```
 tmp=top;
 top=top->next;
 free(tmp);
 return poped;
}

int priority(char symbol)
{if(symbol=='^')
 return 3;
 else if(symbol=='/' || symbol=='*')
 return 2;
 else if(symbol=='+' || symbol=='-')
 return 1;
 else
 return 0;
}

void intopost()
{
 char symbol;
 int i,k=0;
 push('(');
 for(i=0;i<strlen(infix);i++);
 infix[i]=')';
 for(i=0;i<strlen(infix);i++)
 {
 char del;
 symbol=infix[i];
       if((symbol<='9'||(int)symbol<=122)&&(symbol>='0'||(int)symbol>=97))
  postfix[k++]=symbol;
 else if(symbol=='(')
  push(symbol);
 else if(symbol==')')
 {
 while(top->info!='(')
  postfix[k++]=pop();
  del=pop();
     }
 else
 {
while(top!=NULL && (priority(top->info)>=priority(symbol)))
  postfix[k++]=pop();
       push(symbol);
    }
 }
 postfix[k]='\0';

}

void main()
{
 clrscr();
   printf("Enter Infix:");
 gets(infix);
 intopost();
 printf("Postfix is:");
 puts(postfix);
 getch();
}
```

11. Write a program to evaluate postfix expression.

```
#include<stdio.h>
#include<conio.h>
int stack[20];
int top=-1;

void push(int x)
{ stack[++top]=x; }

int pop()
{ return stack[top--]; }

void main()
{char exp[20];
 char *e;
 int n1,n2,n3,num;
 clrscr();
 printf("Enter the Postfix expression :: ");
 scanf("%s",exp);
 e=exp;
 while(*e!='\0')
 { if(isdigit(*e))
   {num=*e-48;
    push(num); }
   else
    {n1=pop();
     n2=pop();
    switch(*e)
    {
     case '+':
     { n3=n1+n2;
        break; }

     case '-':
     { n3=n2-n1;
       break;
       }

      case '*':
      { n3=n1*n2;
       break;
       }

       case '/':
      { n3=n2/n1;
       break;
       }
     }
    push(n3);
    }
   e++;
   }
   printf("\nThe Result of expression %s  =  %d\n\n",exp,pop());
   getch();
}
```