[1] Write c program to find the GCD of two number using recursion.

```
#include<stdio.h>
int gcd(int i,int j)
{
   if(j==0)
      return i;
   else
      return gcd(j,i%j);
}
void main()
{
   int num1,num2,num3,gcd1,g;
   printf("\nEnter 1st positive integer::");
   scanf("%d",&num1);
   printf("\nEnter 2nd positive integer::");
   scanf("%d",&num2);
   printf("\nEnter 3rd positive integer::");
   scanf("%d",&num3);

   g=gcd(num3, gcd(num1, num2));
   printf("\n gcd of %d, %d, %d is: %d\n",num1,num2,num3,g);
}
```

Output:

```
Enter 1st positive integer::3
Enter 2nd positive integer::6
Enter 3rd positive integer::12
gcd of 3, 6, 12 is: 3
```

[2] Write a program to implement linear search.

```
#include<stdio.h>
void linscr(int a[],int n,int x)
{
   int i;
   for(i=0;i<n;i++)
   {
      if(a[i]==x)
      {
         printf("search successful element found at position: %d",i+1);
         break;
      }
   }
   if(i == n)
      printf("search element is unsuccessful");
}
void main()
{
   int a[10],n,i,pos,x;
   printf("enter the array size:");
   scanf("%d",&n);
   printf("enter the array element:");
   for(i=0;i<n;i++)
      scanf("%d",&a[i]);
   printf("enter the array search element:");
   scanf("%d",&x);
   linscr(a,n,x);
}
```

Output 1:

```
enter the array size:5
enter the array element:
1
3
2
6
4
enter the array search element:6
search successful element found at position: 4
```

Output 2:

```
enter the array size:5
enter the array element:
3
2
5
4
6
enter the array search element:8
search element is unsuccessful
```

[3] Write a program to implement Binary search

```
#include<stdio.h>
int binsrch(int a[],int low,int high,int x);
void main()
{
   int a[10],n,i,pos,x;
   printf("\n enter array size:");
   scanf("%d",&n);
   printf("\n enter array elements:");
   for(i=0; i<n; i++)
      scanf("%d",&a[i]);
   printf("\n enter search element:");
   scanf("%d",&x);
   pos=binsrch(a,0,n-1,x);
   if(pos==-1)
      printf("\n search unsuccessful");
   else
      printf("\n search successful at position %d",pos + 1);
}
int binsrch(int a[],int low,int high,int x)
{
   int mid;
   while(low<=high) {
      mid=(low+high)/2;
      if(x < a[mid])
         high=mid-1;
      else if(x > a[mid])
         low=mid+1;
      else
         return mid;
   }
   return -1;
}
```

Output 1:

```
enter array size:5
enter array elements:1 2 3 4 5
enter search element:4
search successful at position 4
```

Output 2:

```
enter array size:5
enter array elements:1 2 3 4 5
enter search element:6
search unsuccessful
```

[4] Write a program to implement bubble sort.

```
#include<stdio.h>
void bubble(int a[],int n);
void main()
{
   int a[10],n,i;
   printf("\n enter array size:");
   scanf("%d",&n);
   printf("\n enter array elements:");
   for(i=0; i<n; i++)
      scanf("%d",&a[i]);
   bubble(a,n);
   printf("\n sorted list:");
   for(i=0; i<n; i++)
      printf("\t%d",a[i]);
}
void bubble(int a[],int n)
{
   int i,j,temp;
   for(i=1; i<n; i++)
   {
      for(j=0; j<n-i; j++)
      {
         if(a[j]>a[j+1])
         {
            temp=a[j];
            a[j]=a[j+1];
            a[j+1]=temp;
         }
      }
   }
}
```

Output:

```
enter array size:5
enter array elements:2 4 1 6 3
sorted list:       1          2          3          4          6
```

[5] Write a program to implement selection sort

```
#include<stdio.h>
void selsort (int a[],int n);
void main()
{
   int a[10],n,i;
   printf("\n enter array size:");
   scanf("%d",&n);
   printf("\n enter array elements:");
   for(i=0; i<n; i++)
      scanf("%d",&a[i]);
   selsort(a,n);
   printf("\n sorted list:");
   for(i=0; i<n; i++)
      printf("\t %d",a[i]);
}
void selsort(int a[],int n)
{
   int i,j,min,pos;
   for(i=0; i<n-1; i++)
   {
      min=a[i];
      pos=i;
      for(j=i+1; j<n; j++)
      {
         if(a[j]<min) {
            min=a[j];
            pos=j;
         }
      }
      a[pos]=a[i];
      a[i]=min;
   }
}
```

Output:

```
enter array size:5
enter array elements:2 4 1 6 3
sorted list:       1          2          3          4          6
```

[6] write a program to check the operations on stack.

```
#include<stdio.h>
#include<stdlib.h>
void push();
void pop();
void display();
int top = -1, stk[10], n = 3;

void main()
{
   int ch, item;
   while(1) {
      printf("\n1> PUSH\t 2> POP\n");
      printf("3> DISPLAY\t 4> exit\n");
      printf("\nEnter the choice: ");
      scanf("%d",&ch);

      switch(ch) {
         case 1: push(); break ;
         case 2: pop(); break;
         case 3: display(); break;
         default: exit(0);
      }
   }
}
void push()
{
   int item;
   if (top==n-1)
   {
      printf("stack full...stack overflow\n");
      return;
   }
   else {
      printf("enter the element \n");
      scanf("%d",&item);
      top++;
      stk[top]=item;
   }
```

```
}
void pop()
{
   int item;
   if(top==-1) {
      printf("empty stack...stack underflow..\n");
      return;
   } else {
      printf("popped element= %d\n", stk[top]);
      top--;
   }
}
void display()
{
   int i;
   if(top==-1)
   {
      printf("empty stack...\n");
      return;
   }
   printf("stack contents are \n");
   for(i=0; i<=top; i++)
      printf(" %d ", stk[i]);
}
```

Output:

```
1> PUSH        2> POP
3> DISPLAY   4> exit

Enter the choice: 1
enter the element
10
1> PUSH        2> POP
3> DISPLAY   4> exit

Enter the choice: 1
enter the element
20
1> PUSH        2> POP
```

```
3> DISPLAY   4> exit

Enter the choice: 1
enter the element
30
1> PUSH        2> POP
3> DISPLAY   4> exit

Enter the choice: 1
stack full...stack overflow

1> PUSH        2> POP
3> DISPLAY   4> exit

Enter the choice: 2
popped element= 30

1> PUSH        2> POP
3> DISPLAY   4> exit

Enter the choice: 2
popped element= 20

1> PUSH        2> POP
3> DISPLAY   4> exit

Enter the choice: 2
popped element= 10

1> PUSH        2> POP
3> DISPLAY   4> exit

Enter the choice: 2
empty stack...stack underflow..

1> PUSH        2> POP
3> DISPLAY   4> exit

Enter the choice: 4
4
```

[7] Write a program to convert an infix expression to postfix expression.

```
#include<stdio.h>
#include<conio.h>
#include<ctype.h>
char stack[100];
int top=-1;
void push(char x)
{
        stack[++top]=x;
}
char pop()
{
        if(top==-1)
                return -1;
        else
                return stack[top--];
}
int priority(char x)
{
        if(x=='(')
                return 0;
        if(x=='+'||x=='-')
                return 1;
        if(x=='*'||x=='/')
                return 2;
        return 0;
}
void main()
{
        char exp[100];
        char *e,x;
        clrscr();
        printf("enter the expression:");
        scanf("%s",exp);
        e=exp;
        while(*e!='\0')
        {
                if(isalnum(*e))
                        printf("%c",*e);
```

```
            else if(*e == '(')
                    push(*e);
            else if(*e==')')
            {
                    while((x=pop())!='(')
                    printf("%c",x);
            }
            else
            {
                    while(priority(stack[top])>=priority(*e))
                            printf("%c",pop());
                    push(*e);
            }
            e++;
    }
    while(top!=-1)
    {
            printf("%c",pop());
    }
    getch();
}
```

Output:

```
enter the expression:(a+b)
ab+
```

[8] Write a program to create a linear queue to insert elements into the list and delete elements from the list.

```
#include<stdio.h>
#include<stdlib.h>
void qinsert();
void qdelete();
void qdisplay();
int queue[10],front=0,rear=-1, max=3;
void main()
{
   int ch;
   while(1) {
      printf("\n LINEAR QUEUE OPERATIONS \n");
      printf("1.Insert \t 2.delete\n");
      printf("3.display\t 4.exit\n");
      printf("enter your choice: ");
      scanf("%d",&ch);
      switch(ch) {
      case 1:
         qinsert();
         break;
      case 2:
         qdelete();
         break;
      case 3:
         qdisplay();
         break;
      case 4:
         exit(0);
      default:
         printf("\n WRONG CHOICE");
      }
   }

}
void qinsert()
{
   int item;
   if(rear==max-1)
```

```
    {
        printf("Queue is full");
        return;
    }
    else {
        printf("enter the value to insert \n");
        scanf("%d",&item);
        rear++;
        queue[rear]=item;
    }
}
void qdelete()
{
    if(front==rear+1)
    {
        printf("Queue is empty");
        return;
    }
    else {
        printf("%d is deleted \n",queue[front]);
        front++;
    }
}
void qdisplay()
{
    int i;
    if(front==rear+1)
    {
        printf("Queue is empty");
        return;
    }
    else
    {
        printf("\n Queue elements \n");
        for(i = front; i <= rear; i++)
            printf("%d\t",queue[i]);
    }
}
```

Output:

```
LINEAR QUEUE OPERATIONS
1.Insert        2.delete
3.display       4.exit
enter your choice: 1
enter the value to insert
10
LINEAR QUEUE OPERATIONS
1.Insert        2.delete
3.display       4.exit
enter your choice: 1
enter the value to insert
20
LINEAR QUEUE OPERATIONS
1.Insert        2.delete
3.display       4.exit
enter your choice: 1
enter the value to insert
30
LINEAR QUEUE OPERATIONS
1.Insert        2.delete
3.display       4.exit
enter your choice: 1
Queue is full
 LINEAR QUEUE OPERATIONS
1.Insert        2.delete
3.display       4.exit
enter your choice: 2
10 is deleted

 LINEAR QUEUE OPERATIONS
1.Insert        2.delete
3.display       4.exit
enter your choice: 2
20 is deleted

 LINEAR QUEUE OPERATIONS
1.Insert        2.delete
3.display       4.exit
```

```
enter your choice: 2
30 is deleted

 LINEAR QUEUE OPERATIONS
1.Insert          2.delete
3.display         4.exit
enter your choice: 2
Queue is empty
 LINEAR QUEUE OPERATIONS
1.Insert          2.delete
3.display         4.exit
enter your choice: 4
```

[9] Write a program to create a circular queue to insert an elements into the list and delete elements from the list, Display your list after every insertion and deletion.

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

int q[10], f=0,r=-1, count=0, n=3;
void insert(int ele);
void deleteq();
void display();

void main()
{
   int ele,ch;
   while(1) {
      printf("\n1.insert \n2.delete \n3.display \n4.exit");
      printf("\nenter your choice");
      scanf("%d",&ch);
      switch(ch) {
      case 1 :
         printf("\n enter ele to insert: ");
         scanf("%d",&ele);
         insert(ele);
         break;
      case 2 :
         deleteq();
         break;
      case 3 :
         display();
         break;
      default :
         exit(0);
      }
   }
}

void insert(int ele)
{
   if(count==n) {
      printf("\nqueue is full");
```

```
        return;
    }
    count++;
    r=(r+1)%n;
    q[r]=ele;
    display();
}

void deleteq()
{
    if(count==0) {
        printf("\nqueue is empty");
        return;
    }
    printf("\ndeleted element=%d",q[f]);
    f=(f+1)%n;
    count--;
    if(count == 0) {
        f=0;
        r=-1;
    }
    display();
}

void display()
{
    int i, j;
    if(count==0) {
        printf("\nqueue is empty");
        return;
    }
    printf("\nQueue elements are");
    j=f;
    for(i=0; i< count; i++) {
        printf(" %d ", q[j]);
        j=(j+1)%n;
    }
}
```

Output:

```
1.insert      2.delete
3.display   4.exit
enter your choice 1
enter ele to insert: 10
Queue elements are 10

1.insert      2.delete
3.display   4.exit
enter your choice 1
enter ele to insert: 20
Queue elements are 10  20

1.insert      2.delete
3.display   4.exit
enter your choice 1
enter ele to insert: 30
Queue elements are 10  20  30

1.insert      2.delete
3.display   4.exit
enter your choice 2
deleted element=10
Queue elements are 20  30

1.insert      2.delete
3.display   4.exit
enter your choice 1
enter ele to insert: 40
Queue elements are 20  30  40

1.insert      2.delete
3.display   4.exit
enter your choice 4
```