1. **Write a C++ program to read one dimensional array and print sum of all elements along with inputted array elements using dynamic memory allocation.**

```
#include<iostream.h>
int main()
{
int n,sum=0;
cout<<"Enter the size of the array:";
cin>>n;
int*arr=new int[n];
cout<<"Enter the array elements:";
for(int i=0;i<n;i++)
{
cin>>arr[i];
sum+=arr[i];
}
{
cout<<arr[i]<<" ";
}
cout<<"\nSum of all elements:"<<sum<<endl;
return 0;
}
```

Output:-

Enter the size of the array:5

Enter the array elements:

3

3

4

5

6

Sum of all the elements:21

2. **Write a C++ program to find GCD using recursive function.**

```
#include<iostream.h>
int gcd(int a, int b)
{
if(b==0)
return a;
else
return gcd(b,a%b);
}
int main()
{
int num1,num2;
cout<<"Enter first number:";
cin>>num1;
cout<<"Enter second number:";
cin>>num2;
cout<<"GCD of "<<num1<<" and "<<num2<<"
is:"<<gcd(num1,num2)<<endl;
return 0;

}
```

Output:-

Enter first number: 24

Enter second number: 16

GCD of 24 and 16 is: 8

3. **Write a C++ program to display Pascal triangle using binomial function.**

```
#include<iostream.h>

int binomialCoeff(int n, int k)

{

if(k>n-k)k=n-k;

int res=1;

for(int i=0;i<k;i++)

{

res*=(n-i)

res/=(i+1);

}

return res;

}

void printPascalTriangle(int rows)

{

for(int n=0;n<rows;n++)

{

for(int s=0;s<rows-n-1;s++)cout<<" ";

for(int k=0;k<=n;k++)

{

cout<<binomialCoeff(n,k)<<" ";

}

cout<<endl;

}

}
```

```
int main()

{

int n;

cout<<"Enter number of rows:";

cin>>n;

printPascalTriangle(n);

return 0;

}
```

Output:-

```
Enter number of rows: 5

                    1

                1       1

            1       2       1

        1       3       3       1

    1       4       6       4       1
```

4. **Write a C++ program to generate n Fibonacci numbers using recursive function.**

```
#include<iostream.h>
int fibo(int n)
{
if(n==0)
return 0;
else if(n==1)
return 1;
else
return fibo(n-1)+fibo(n-2);
}
int main()
{
int n;
cout<<"Enter the value of n for the fibonacci series:";
cin>>n;
cout<<"Fibonacci series up to "<<n<<":";
for(int i=0;i<n;i++)
{
cout<<fibo(i)<<" ";
}
```

```
return 0;

}
```

Output:-

```
Enter the value of n Fibonacci series: 6

Fibonacci series up to 6: 0 1 1 2 3 5
```

5. Write a C++ program to implement Tower of Hanoi.

```
#include<iostream.h>
void ToH(int n, char from_rod, char to_rod,char aux_rod)
{
if(n==1)
{
cout<<"Move disk 1 from rod "<<from_rod<<" to rod "<<to_rod<<endl;
return;
}
ToH(n-1,from_rod,aux_rod,to_rod);
cout<<"Move disk "<<n<<" from rod "<<from_rod<<" to rod
"<<to_rod<<endl;
ToH(n-1,aux_rod,to_rod,from_rod);
}
int main()
{
int n;
cout<<"Enter number of disks:";
cin>>n;
ToH(n,'A','C','B');
return 0;
}
```

Output:-

```
Enter number of disks: 3
```

Move disk 1 from rod A to rod C

Move disk 2 from rod A to rod B

Move disk 1 from rod C to rod B

Move disk 3 from rod A to rod C

Move disk 1 from rod B to rod A

Move disk 2 from rod B to rod C

Move disk 1 from rod A to rod C

6. **Write a C++ program to read the names of cities and arrange them alphabetically.**

```
#include<iostream.h>
#include<stdio.h>
#include<string.h>
int main()
{
int n;
cout<<"Enter number of cities:";
cin>>n;
char cities[100][50];
cout<<"Enter city names:\n";
for(i=0;i<n;i++)
{
cin>>cities[i];
}
for(i=0;i<n-1;i++)
{
for(int j=i+1;j<n;j++)
{
if(strcmp(cities[i], cities[j])>0)
{
```

```
        char temp[50];
        strcpy(temp,cities[i]);
        strcpy(cities[i],cities[j]);
strcpy(cities[j],temp);
}
}
}
cout<<"\nCities in alphabetical order:\n";
for(i=0;i<n;i++)
{
cout<<cities[i]<<endl;
}
return 0;
}
```

Output:-

```
Enter number of cities:5
Enter city names:
Delhi
Mumbai
Kolkata
Chennai
Bangalore
Cities in alphabetical order:
Bangalore
```

Chennai

Delhi

Kolkata

Mumbai

7. Write a C++ program to delete repeated elements .

```
#include<iostream.h>
int main()
{
int n;
cout<<"Enter the size of the array:";
cin>>n;
int arr[n];
cout<<"Enter "<<n<<" elements:";
for(int i=0;i<n;i++)
{
cin>>arr[i];
}
int newSize=n;
```

```
        for(i=0;i<newSize;i++)
        {
        for(int j=i+1;j<newSize;)
        {
        if(arr[i]==arr[j])
        {
        for(int k=j;k<newSize-1;k++)
        {
arr[k]=arr[k+1];
}
newSize--;
}
Else
{
j++;
}
}
}
cout<<"Array after removing duplicates:";
for(i=0;i<newSize;i++)
{
cout<<arr[i]<<" ";
}
cout<<endl;
return 0;
}
```

Output:-

Enter the size of the array: 8

Enter 8 elements: 4 5 2 4 3 2 1 5

Array after removing duplicates: 4 5 2 3 1

8. Write a C++ program to search an element using linear search technique.

```
#include<iostream.h>
    int main()
    {
    int val, n1, a[10];
    cout<<"Enter number of elements in list:";
```

```
cin>>n1;
cout<<"\nEnter the array elements:\n";
for(int i=0;i<n1;i++)
{
cin>>a[i];
}
cout<<"\nEnter the value to be searched:";
cin>>val;
for(i=0;i<n1;i++)
{
if(a[i]==val)
{
cout<<"\nElement is present at position "<<(i+1);
return 0;
}
}
cout<<"\nElements is not present in the array.";
return 0;
}
```

Output1:-

```
Enter number of elements in list: 5
Enter the array elements:
10
20
30
40
50
Enter the value to be searched: 50
Element is present at position 5
```

Output2:-

```
Enter number of elements in list: 5
Enter the array elements:
10
20
30
40
```

50

Enter the value to be searched: 60

Element is not present in the array

.

9. **Write a C++ program to search an element using recursive binary search technique.**

```
#include<iostream.h>
int bsearchRec(int lb,int ub, int a[], int key)
{
if(lb>ub)
return -1;
int mid=(lb+ub)/2;
if(a[mid]==key)
return mid;
else if(a[mid]<key)
return bsearchRec(mid+1,ub,a,key);
else
return bsearchRec(lb,mid-1,a,key);
}
int main()
{
int n,key;
cout<<"Enter the number of elements:";
cin>>n;
int*a=new int[n];
cout<<"Enter the sorted elements:";
for(int i=0;i<n;i++)
cin>>a[i];
cout<<"Enter the key to be searched:\n";
cin>>key;
int result=bsearchRec(0,n-1,a,key);
if(result==-1)
cout<<"Element not found!"<<endl;
else
cout<<"Element found at position:"<<result+1<<endl;
return 0;
}
```

Output1:-

```
Enter the number of elements: 5
Enter the sorted elements:
2
5
8
12
30
Enter the key to be searched: 8
Element found at position: 3
```

Output2:-

```
Enter the number of elements: 5
Enter the sorted elements:
2
5
8
12
30
Enter the key to be searched: 9
Element not found!.
```

Part –B

1. **Write a C++ program to sort the given list using selection sort technique.**

```
#include<iostream.h>
void selectionSort(int arr[],int n)
{
for(int i=0;i<n-1;i++)
{
int minIndex=i;
for(int j=i+1;j<n;j++)
{
if(arr[j]<arr[minIndex])
{
minIndex=j;
```

```
}
}
int temp=arr[i];
arr[i]=arr[minIndex];
arr[minIndex]=temp;
}
}
void printArray(int arr[],int n)
{
for(int i=0;i<n;i++)
{
cout<<arr[i]<<",";
}
cout<<endl;
}
int main()
{
int n;
cout<<"Enter number of elements:";
cin>>n;
int*arr=new int[n];
cout<<"Enter "<<n<<" elements:";
for(int i=0;i<n;i++)
{
cin>>arr[i];
}
cout<<"Original array:";
printArray(arr,n);
selectionSort(arr,n);
cout<<"Sorted array:";
printArray(arr,n);
return 0;
}
```

Output:-

```
Enter number of elements: 5
```

Enter 5 elements:
12
4
65
9
1
Original array: 12, 4, 65, 9, 1
Sorted array: 1, 4, 9, 12, 65

2. **Write a C++ program to sort the given list using insertion sort technique.**

```
#include<iostream.h>
void insertionSort(int arr[],int n)
{
for(int i=1;i<n;i++)
{
int key=arr[i];
int j=i-1;
while(j>=0&&arr[j]>key)
{
arr[j+1]=arr[j];
j--;
}
arr[j+1]=key;
}
}
void printArray(int arr[],int n)
{
for(int i=0;i<n;i++)
cout<<arr[i]<<",";
cout<<endl;
}
int main()
{
```

```
int n;
cout<<"Enter number of elements:";
cin>>n;
int*arr=new int[n];
cout<<"Enter "<<n<<" elements:";
for(int i=0;i<n;i++)
{
cin>>arr[i];
}
cout<<"Original array:";
printArray(arr,n);
insertionSort(arr,n);
cout<<"Sorted array:";
printArray(arr,n);
return 0;
}
```

Output:-

```
Enter number of elements: 7
Enter 7 elements:
12
3
0
44
2
99
5
Original array: 12, 3, 0, 44, 2, 99, 5
Sorted array: 0, 2, 3, 5, 12, 44, 99
```

3. **Write a C++ program to sort the given list using merge sort technique.**

```
#include<iostream.h>
void MergeSort(int a[],int low,int high);
```

```
void Merge(int a[],int low,int mid,int high);
int main()
{
int a[100];
int n;
cout<<"Enter array size:";
cin>>n;
cout<<"Enter the array:\n";
for(i=0;i<n;i++)
cin>>a[i];
MergeSort(a,0,n-1);
cout<<"The sorted array is:\n";
for(i=0;i<n;i++)
cout<<a[i]<<" ";
cout<<endl;
return 0;
}
void MergeSort(int a[],int low,int high)
{
if(low<high)
{
int mid=(low+high)/2;
MergeSort(a,low,mid);
MergeSort(a,mid+1,high);
Merge(a,low,mid,high);
}
}
void Merge(int a[],int low,int mid,int high)
{
int b[100];
int i=low,h=low,j=mid+1;
while(h<=mid&&j<=high)
{
if(a[h]<a[j])
b[i++]=a[h++];
else
```

```
b[i++]=a[j++];
}
while(h<=mid)
b[i++]=a[h++];
while(j<=high)
b[i++]=a[j++];
for(int k=low;k<=high;k++)
a[k]=b[k];
}
```

Output:-

```
Enter array size: 8
Enter the array:
5 2 8 1 9 12 11 10
The sorted array is:
1 2 5 8 9 10 11 12
```

4. **Write a C++ program to implement stack.**

```
#include<iostream.h>
#include<stdlib.h>
int main()
{
int s[10],top=0,ch,item;
do
{
cout<<"\n1. PUSH\n";
cout<<"2. POP\n";
cout<<"3. DISPLAY\n";
cout<<"4. EXIT\n";
cout<<"Enter your choice:";
cin>>ch;
switch(ch)
{
case 1:
if(top>=10)
{
```

```
cout<<"Stack is full\n";
}
else{
cout<<"Enter item to push:";
cin>>item;
top++;
s[top]=item;
}
break;
case 2:
if(top==0)
{
cout<<"Stack is empty\n";
}
else{
cout<<"Deleted item="<<s[top]<<endl;
top--;
}
break;
case 3:
if(top==0)
{
cout<<"Stack is empty\n";
}
else{
cout<<"Stack elements:\n";
for(int i=top;i>=1;i--)
{
cout<<"s["<<i<<"]="<<s[i]<<endl;
}
}
break;
case 4:
exit(0);
default:
cout<<"Invalid choice.\n";
```

```
}
}while(ch!=4);
return 0;
}
```

Output:-

1. **PUSH**
2. **POP**
3. **DISPLAY**
4. **EXIT**

Enter your choice: 1

Enter item to push: 10

1. **PUSH**
2. **POP**
3. **DISPLAY**
4. **EXIT**

Enter your choice: 1

Enter item to push: 20

1. **PUSH**
2. **POP**
3. **DISPLAY**
4. **EXIT**

Enter your choice: 1

Enter item to push: 30

1. **PUSH**
2. **POP**
3. **DISPLAY**
4. **EXIT**

Enter your choice: 3

Stack elements:

S[3]=30

S[2]=20

S[1]=10

1. PUSH
2. POP
3. DISPLAY
4. EXIT

Enter your choice: 2

Deleted item=30

1. PUSH
2. POP
3. DISPLAY
4. EXIT

Enter your choice: 3

Stack elements:

S[2]=20

S[1]=10

1. PUSH
2. POP
3. DISPLAY
4. EXIT

Enter your choice: 4

5. Write a C++ program to convert an infix expression to postfix.