PART B

9.) Write a C++ program to create a class string with default parameterized and copy constructor. Implement the following by overloading operation :- strcpy(),strcmp(),strcat().

```
#include<iostream>
#include<string>
using namespace std;
class strings
{
char *str;

public:
strings(){}
strings(char s)
{

Str=s;
}
```

```
void operator=(strings);
int operator==(strings);
strings operator+(strings);
friend ostream& operator<<(ostream&,strings);
};
void strings::operator=(strings s2)
{
int i=0;
while(s2.str[i]!='\0')
{
str[i]=s2.str[i];
i++;
}
str[i]='\0';
}
strings strings::operator+(strings s2)
{
int i=0,j=0;
while(str[i]!='\0')
i++;
while(s2.str[j]!='\0')
{
str[i]=s2.str[j];
i++;
j++;
}
str[i]='\0';
return *this;
}
int strings::operator==(strings s2)
{
int i=0;
while(str[i]!='\0'||s2.str[i]!='\0')
{
if(str[i]!=s2.str[i])
return(str[i]-s2.str[i]);
i++;
}
```

```
return 0;
}
ostream& operator<<(ostream& os,strings s)
{
cout<<s.str;
return os;
}
int main()
{
char str1[80],str2[80];
cout<<"\nEnter 2 strings:\n";
cin>>str1>>str2;
strings s1(str1),s2(str2);
strings t1(s1),t2=s2;
if((s1==s2)==0)
cout<<"\nStrings are equal\n";
else if((s1==s2)>0)
cout<<"\nString 1 is greater\n";
else
cout<<"\nString 2 is greater\n";
strings s3=s1+s2;
cout<<"\nAfter concatenation\n1st string= "<<s3<<"\n2nd string= "<<s2;
t1=t2;
cout<<"\nAfter copying\n1st string= "<<t1<<"\n2nd string= "<<t2<<"\n";
return 0;
}
```

10.) Write a C++ program to implement following operator overloading concept using complex number. + ,- ,+ +, -, = =

```
#include<iostream>
using namespace std;
class comp
{
int r,i;
public:
void scan()
{
cout<<"Enter real & imaginary part of complex no. \n";
cin>>r>>i;
}
void sd()
{
```

```
if(i<0)
cout<<"complex no. is "<<r<<i<<"i \n";
else
cout<<"complex no. is "<<r<<"+"<<i<<"i \n";
}
comp operator+(comp x);
comp operator-(comp x);
friend comp operator++(comp &a);
friend comp operator+(int a,comp &x);
friend comp operator+(comp &x,int b);
comp operator=(comp p);
};
comp comp::operator+(comp x)
{
comp a;
a.r=x.r+r;
a.i=x.i+i;
return a;
}
comp comp::operator-(comp x)
{
comp b;
b.r=r-x.r;
b.i=i-x.i;
return b;
}
comp operator++(comp &a)
{
a.r++;
return a;
}
comp operator+(int a,comp &x)
{
// comp p;
x.r=a+x.r;
// return p;
}
comp operator+(comp &x,int b)
{
x.r=x.r+b;
// return q;
}
comp comp::operator=(comp p)
```

```
{
r=p.r;
i=p.i;
```

```
return *this;
}
int main()
{
comp a,b,c;
int x;
a.scan();
b.scan();
a.sd();
b.sd();
cout<<"Adding two complex no's \n";
(a+b).sd();
cout<<"Subtracting two complex no's \n";
(a-b).sd();
c=a;
cout<<"Assign obj a to obj c \n";
c.sd();
operator++(a);
cout<<"After incrementing real part of a \n";
a.sd();
cout<<"enter an integer \n";
cin>>x;
cout<<"Adding integer to complex no. \n";
operator+(x,a);
a.sd();
operator+(c,x);
c.sd();
return 0;
}
```

11.) Write a C++ program to sort integers and floating point numbers using template.

```
#include<iostream>
using namespace std;
template<class x > void sort(x *a,int size)
{
x t;
for(int i=0;i<size;i++)
for(int j=(size-1);j>=i;j--)
{
if(a[j-1]>a[j])
{
t=a[j-1];
```

```
a[j-1]=a[j];
a[j]=t;
```

```
}
}
}
int main()
{
int i,n;
cout<<"enter integer array size\n";
cin>>n;
int p[n];
cout<<"Enter "<<n<<" integer array elements\n";
for(i=0;i<n;i++)
cin>>p[i];
cout<<"Actual integer array before sorting is:\n";
for(i=0;i<n;i++)
cout<<p[i]<<" ";
cout<<"\n";
sort(p,n);
cout<<"Sorted Integer array is:\n";
for(i=0;i<n;i++)
cout<<p[i]<<" ";
cout<<"\n";
char q[n];
cout<<"Enter "<<n<<" character array elements\n";
for(i=0;i<n;i++)
{cin>>q[i];}
cout<<"Actual character array before sorting is:\n";
for(i=0;i<n;i++)
cout<<q[i]<<" ";
cout<<"\n";
sort(q,n);
cout<<"Sorted character array is:\n";
for(i=0;i<n;i++)
cout<<q[i]<<" ";
cout<<"\n";
float r[n];
cout<<"Enter "<<n<<" float array elements\n";
for(i=0;i<n;i++)
cin>>r[i];
cout<<"Actual float array before sorting is:\n";
for(i=0;i<n;i++)
cout<<r[i]<<" ";
cout<<"\n";
sort(r,n);
```

```
cout<<"Sorted float array is:\n";
for(i=0;i<n;i++)
cout<<r[i]<<”\n”;
return 0;
}
```