Program 1

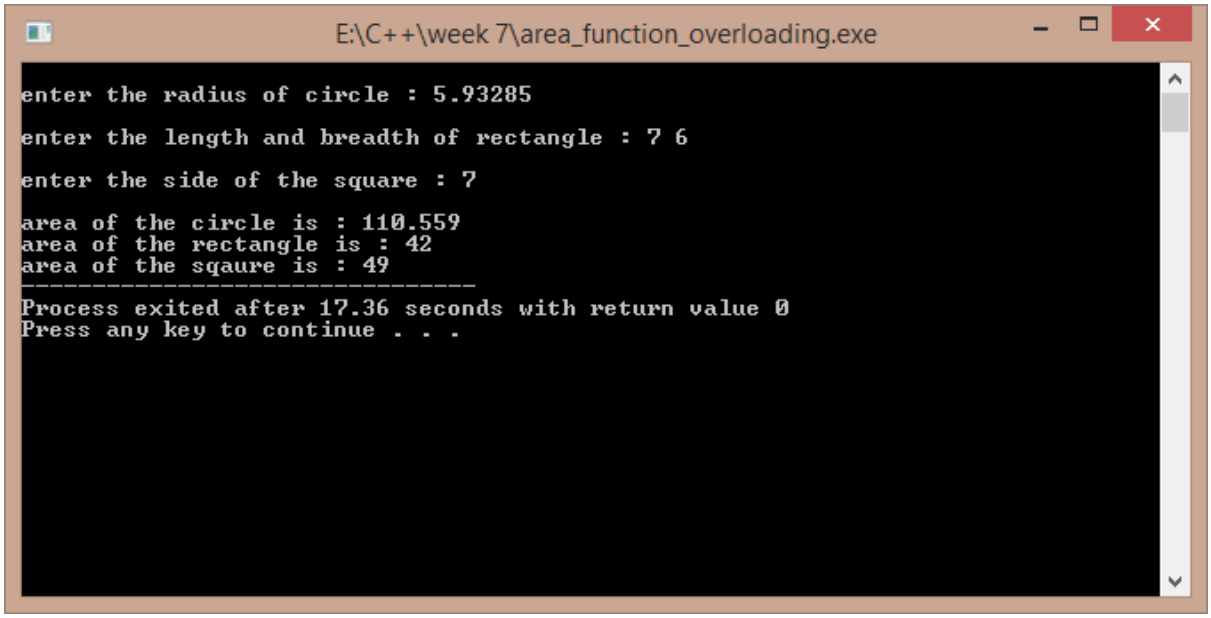
//a program to depict the function overloading concept

```
#include<iostream>
using namespace std;
float area(float r)
{
	return (3.141)*r*r;
}
int area(int a,int b)
{
	return a*b;
}
int area(int s)
{
	return s*s;
}
int main()
{
	float r,ac;
	int a,b,s,ar,as;
	cout<<"\nenter the radius of circle : ";
```

```
        cin>>r;
        ac=area(r);
        cout<<"\nenter the length and breadth of rectangle : ";
        cin>>a>>b;
        ar=area(a,b);
        cout<<"\nenter the side of the square : ";
        cin>>s;
        as=area(s);
        cout<<"\narea of the circle is : "<<ac;
        cout<<"\narea of the rectangle is : "<<ar;
        cout<<"\narea of the sqaure is : "<<as;
return 0;

}
```

Output:-



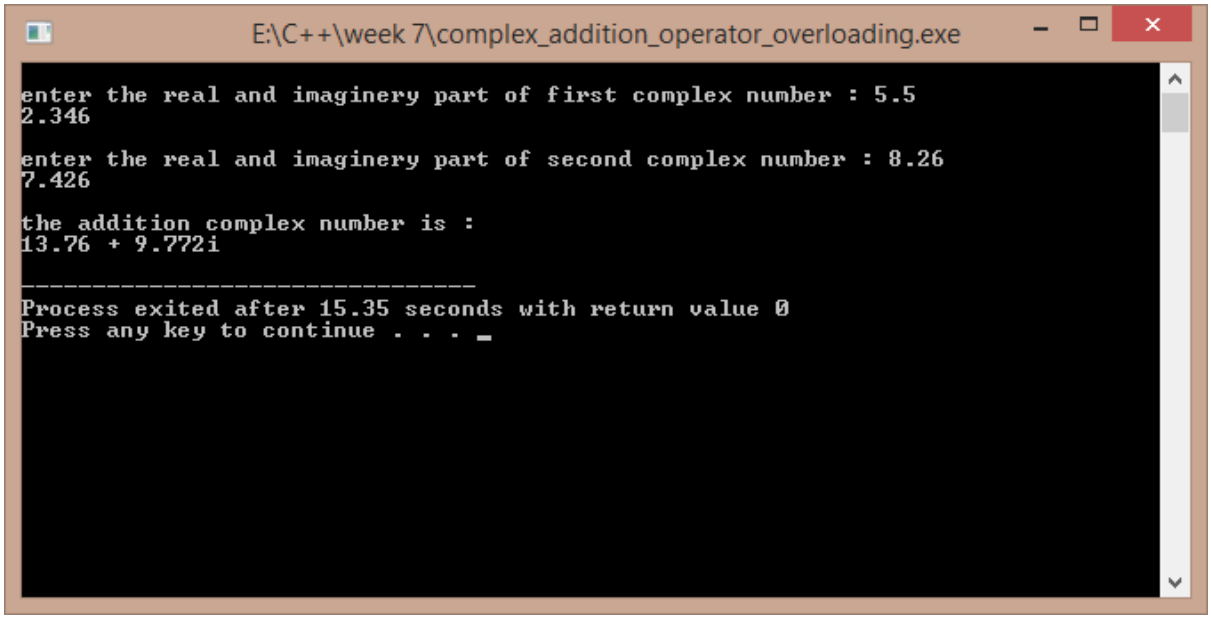
Program 2

```
//a program to add complex numbers using operator overloading
#include<iostream>
using namespace std;
class complex1
{
	float real,imag;
	public:
		complex1()
		{
			real=imag=0;
		}
		complex1(float a,float b)
		{
```

```
                real=a;
                imag=b;
        }
        void display();
        complex1 operator +(complex1 m);
};
void complex1::display()
{
    cout<<"\n"<<real<<" + "<<imag<<"i"<<endl;
}
complex1 complex1::operator +(complex1 m)
{
    complex1 temp;
    temp.real=this->real+m.real;
    temp.imag=this->imag+m.imag;
    return temp;
}
int main()
{
    complex1 a,b;
    float p,q,r,s;
    cout<<"\nenter the real and imaginery part of first complex number : ";
    cin>>p>>q;
    cout<<"\nenter the real and imaginery part of second complex number : ";
```

```
        cin>>r>>s;
        a=complex1(p,q);
        b=complex1(r,s);
        complex1 c;
        c=a+b;
        cout<<"\nthe addition complex number is : ";
        c.display();
return 0;
}
```

Output:-



Program 3

```
//a program to overload the operator regarding the strings
#include<iostream>
```

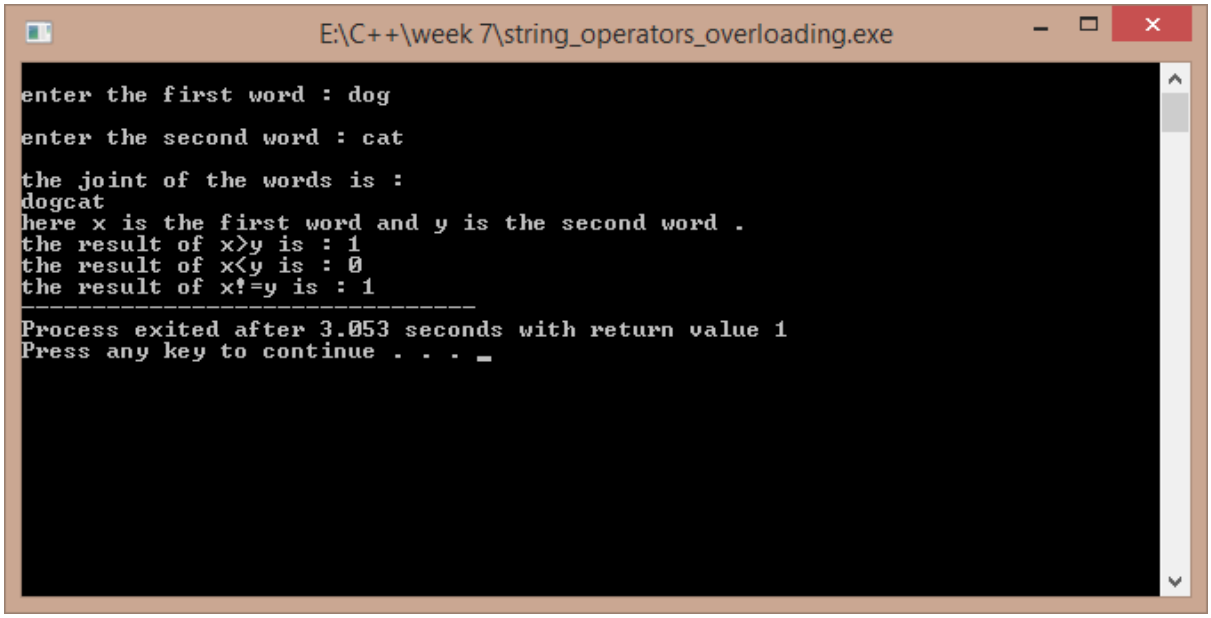
```
#include<cstring>
using namespace std;
class str
{
	char *s;
	int l;
	public:
		str()
		{
			l=0;
			s=new char[l+1];
		}
		str(char *p)
		{
			l=strlen(p);
			s=new char[l+1];
			strcpy(s,p);
		}
		str(int k)
		{
			l=k;
			s=new char[l+1];
		}
		str operator +(str);
```

```
        int operator <(str);
        int operator >(str);
        void operator =(str);
        int operator !=(str);
        void display()
        {
        //      cout<<this->l;
                cout<<"\n"<<this->s;
        }
};
str str::operator +(str a)
{
        str t;
        t=str(strlen(this->s)+strlen(a.s));
        strcpy(t.s,this->s);
        strcat(t.s,a.s);
        return t;
}
int str::operator >(str a)
{
        if(strcmp(this->s,a.s)>0)
        return 1;
        else
        return 0;
```

```
}
int str::operator <(str a)
{
        if(strcmp(this->s,a.s)<0)
        return 1;
        else
        return 0;
}
void str::operator =(str a)
{
        strcpy(this->s,a.s);
}
int str::operator !=(str a)
{
        if(strcmp(this->s,a.s)!=0)
        return 1;
        else
        return 0;
}
int main()
{
        str x,y;
        char temp[100];
```

```
        cout<<"\nenter the first word : ";
        cin>>temp;
        x=str(temp);
        cout<<"\nenter the second word : ";
        cin>>temp;
        y=str(temp);
        str xy;
        xy=x+y;
        cout<<"\nthe joint of the words is : ";
        xy.display();
        cout<<"\nhere x is the first word and y is the second word .";
        cout<<"\nthe result of x>y is : "<<(x>y);
        cout<<"\nthe result of x<y is : "<<(x<y);
        cout<<"\nthe result of x!=y is : "<<(x!=y);
        return 1;
}
```

Output :-


E:\C++\week 7\string_operators_overloading.exe
enter the first word : dog
enter the second word : cat
the joint of the words is :
dogcat
here x is the first word and y is the second word .
the result of x>y is : 1
the result of x<y is : 0
the result of x!=y is : 1
Process exited after 3.053 seconds with return value 1
Press any key to continue . . .