**Practical No.2**

**Using friend function**

**Program 2(a): Write a friend function for adding for adding the two complex numbers, using a single class.**

**Example:** Write a program to add two complex numbers using a friend function to add the complex numbers.

**Coding:**

#include<iostream.h>

#include<conio.h>

class Complex

{

int x,y;

public:

void read()

{

cout<<"Enter the real and imaginary parts of a complex number:";

cin>>x>>y;

}

friend Complex add(Complex c1, Complex c2);

void display()

{

if(y<0)

cout<<x<<y<<"i";

else

cout<<x<<"+i"<<y;

}

};

Complex add(Complex c1, Complex c2)

{

Complex c;

c.x=c1.x+c2.x;

c.y=c1.y+c2.y;

return c;

}

void main()

{

clrscr();

Complex c1;

Complex c2;

Complex c3;

c1.read();

c2.read();

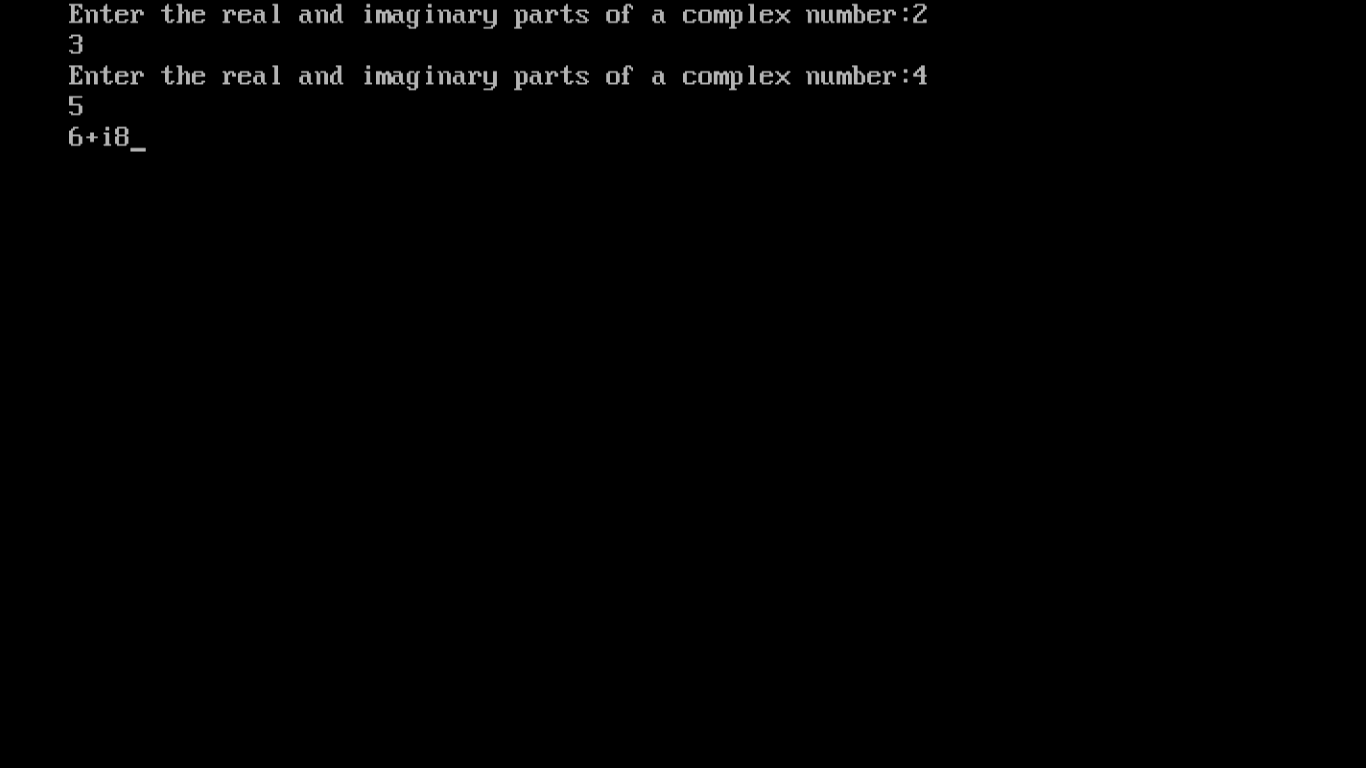
c3=add(c1,c2);

c3.display();

getch();

}

**Output:**



**Program 2(b): Write a friend function for adding the two different distances and display its sum, using two classes.**

**Example:** Write a program to add two distances entered by the user in feet and inches using friend function.

**Coding:**

#include<iostream.h>

#include<conio.h>

class Distance

{

int ft,in;

public:

void read()

{

cout<<"Enter the distance in feet and inches:";

cin>>ft>>in;

}

friend Distance add(Distance d1, Distance d2);

void display()

{

cout<<"Distance is "<<ft<<" feet and "<<in<<" inches.";

}

};

Distance add(Distance d1, Distance d2)

{

Distance dr;

dr.ft=d1.ft+d2.ft;

dr.in=d1.in+d2.in;

if(dr.in>=12)

{

dr.ft++;

dr.in=dr.in-12;

}

return dr;

}

void main()

{

clrscr();

Distance d1;

Distance d2;

Distance d3;

d1.read();

d2.read();

d3=add(d1,d2);

d3.display();

getch();

}

**Output:**



**Program 2(c):** Write a friend function for adding the two matrix from two different classes and display its sum.

**Coding:**

#include<iostream.h>

#include<conio.h>

class Matrix2;

class Matrix1

{

int i,j,m,n,a[10][10];

public:

Matrix1(int x, int y)

{

m=x;

n=y;

}

void readMatrix()

{

for(i=0;i<=m-1;i++)

{

for(j=0;j<=n;j++)

{

cout<<"Enter a number:";

cin>>a[i][j];

}

}

}

void display()

{

for(i=0;i<=m-1;i++)

{

for(j=0;j<=n-1;j++)

{

cout<<a[i][j]<<"\t";

}

cout<<endl;

}

}

friend Matrix1 add(Matrix1 m1, Matrix2 m2, int m, int n);

};

class Matrix2

{

int i,j,m,n,a[10][10];

public:

Matrix2(int x, int y)

{

m=x;

n=y;

}

void readMatrix()

{

for(i=0;i<=m-1;i++)

{

for(j=0;j<=n-1;j++)

{

cout<<"Enter a number:";

cin>>a[i][j];

}

}

}

void display()

{

for(i=0;i<=m-1;i++)

{

for(j=0;j<=n-1;j++)

{

cout<<a[i][j]<<"\t";

}

cout<<endl;

}

}

friend Matrix1 add(Matrix1 m1, Matrix2 m2, int m, int n);

};

Matrix1 add(Matrix1 m1, Matrix2 m2, int m, int n)

{

int i,j;

Matrix1 m3(m,n);

for(i=0;i<=m-1;i++)

{

for(j=0;j<=n-1;j++)

{

m3.a[i][j]=m1.a[i][j]+m2.a[i][j];

}

}

return m3;

}

void main()

{

int m,n;

clrscr();

cout<<"Enter number of rows and columns:";

cin>>m>>n;

Matrix1 m1(m,n);

Matrix2 m2(m,n);

m1.readMatrix();

m2.readMatrix();

Matrix1 m3=add(m1,m2,m,n);

m3.display();

getch();

}

**Output:**

