```
#include <iostream>

using namespace std;

class Number2;

class Number1
{
   int num;
public:
   void get_num()
   {
      cout<<"Enter number:"<<endl;
      cin>> num;
   }
   void show_num()
   {
      cout<<"Numbers is: "<<num<<endl;
   }

   friend void swap_num(Number1 &,Number2 &);
};

class Number2
{
   int num;
public:
   void get_num()
```

```
    {
        cout<<"Enter number:"<<endl;
        cin>> num;
    }
    void show_num()
    {
        cout<<"Numbers is: "<<num<<endl;
    }

    friend void swap_num(Number1 &,Number2 &);
};

void swap_num(Number1 &n1, Number2 &n2)
{
    int temp;
    temp=n2.num;
    n2.num=n1.num;
    n1.num=temp;

}

int main()
{
    Number1 n1;
    Number2 n2;
    n1.get_num();
    n2.get_num();
    swap_num(n1,n2);
```

```
    n1.show_num();
    n2.show_num();

    return 0;
}



#include <iostream>

using namespace std;

class us_currency;

class nepal_currency
{
    float rs;

public:
    void read_rs()
    {
        cout<<"Enter currency(Rs):"<<endl;
        cin>>rs;
    }

    void show_rs()
    {
        cout<<"currency(Rs): "<<rs<<endl;
```

```
    }

    friend bool operator >(nepal_currency,us_currency);
    friend bool operator ==(nepal_currency,us_currency);

};

class us_currency
{
    float dollar;
public:
    void read_usd()
    {
        cout<<"Enter currency(USD):"<<endl;
        cin>>dollar;
    }

    void show_us()
    {
        cout<<"currency(USD): "<<dollar<<endl;

    }

    friend bool operator >(nepal_currency,us_currency);
    friend bool operator ==(nepal_currency,us_currency);
};
```

```
bool operator >(nepal_currency np, us_currency usd)
{

   if((usd.dollar*101.36)<np.rs)
      return 1;
   else
      return 0;

}

bool operator ==(nepal_currency np,us_currency usd)
{

   if((float)(usd.dollar*101.36)==(float)np.rs)
      return 1;
   else
      return 0;
}

int main()
{
   nepal_currency np;
   us_currency usd;
   np.read_rs();
   usd.read_usd();

   if(np==usd)
      cout<<"Both are equal!"<<endl;
```

```
    else if(np>usd)
        cout<<"RS is greater than USD!"<<endl;
    else
        cout<<"USD is greater than RS!"<<endl;

    return 0;
}
```

```
#include <iostream>

using namespace std;

class Complex
{
    float rel, img;
public:
    Complex(float r,float i)
    {
        rel=r;
        img=i;
    }
    Complex()
    {

    }
```

```
    void show()
    {
        cout<<"Real Part: "<<rel<<endl;
         cout<<"Imaginary Part: "<<img<<endl;
    }

    friend Complex operator +(Complex,Complex);
    friend Complex operator +(Complex,float);
    friend Complex operator -(Complex,Complex);
    friend Complex operator *(Complex,Complex);
    friend Complex operator /(Complex,Complex);

};

Complex operator +(Complex c1,Complex c2)
{
    Complex temp;
    temp.rel=c2.rel+c1.rel;
    temp.img=c2.img+c1.img;

    return temp;
}

Complex operator -(Complex c1,Complex c2)
{
    Complex temp;
    temp.rel=c2.rel-c1.rel;
```

```
    temp.img=c2.img-c1.img;

    return temp;
}

Complex operator +(Complex c1,float f)
{
    Complex temp;
    temp.rel=c1.rel+f;
    temp.img=c1.img+f;

    return temp;
}

Complex operator *(Complex c1,Complex c2)
{

    //for complex multiplication (x+yi)(u+vi)=(x*u-y*v)+(x*v+y*u)i

    Complex temp;
    temp.rel=((c1.rel*c2.rel)-(c1.img*c2.img));
    temp.img=((c1.rel*c2.img)+(c1.img*c2.rel));

    return temp;
}
Complex operator /(Complex c1,Complex c2)
{
    //for complex division (a+ib)/(c+id)=((ac+bd)/(cc+dd))+((bc-ad)/(cc+dd))i
```

```
    Complex temp;
    temp.rel=(((c1.rel*c1.rel)+(c1.img*c2.img)) / ((c2.rel*c2.rel)+(c2.img*c2.img)));
    temp.img=(((c1.img*c2.rel)-(c1.rel*c2.img)) / ((c2.rel*c2.rel)+(c2.img*c2.img)));

    return temp;
}

int main()
{
    Complex c1(1,-1),c2(1,1),c3,c4,c5,c6,c7;

    c3=c1+c2;
    c4=c1-c2;
    c5=c1+100;
    c6=c1*c2;
    c7=c1/c2;

    /*c3.show();
    c4.show();
    c5.show();
    c6.show();*/
    c7.show();
    return 0;
}
```

```
#include <iostream>

using namespace std;

class Time
{
   int hr,mi,sec;

public:
   Time(int H, int M, int S)
   {
      hr=H;
      mi=M;
      sec=S;
   }
   Time()
   {
      hr=0;
      mi=0;
      sec=0;
   }

   void show()
   {
      cout<<hr<<" : "<<mi<<" : "<<sec<<endl;
   }

   friend Time operator +(Time,Time);
```

```
    friend Time operator -(Time,Time);
    friend bool operator >(Time,Time);
};

Time operator +(Time t1, Time t2)
{
    Time t;
    t.sec=t1.sec+t2.sec;
    t.mi=t1.mi+t2.mi+(t.sec/60);
    t.hr=t1.hr+t2.hr+(t.mi/60);
    t.sec=t.sec%60;
    t.mi=t.mi%60;

    return t;
}

Time operator -(Time t1, Time t2)
{

    Time t;
    int T, T1, T2;
    /*if(t1.sec<t2.sec)
    {
        t1.sec=t1.sec+60;
        t1.mi=t1.mi-1;
        t.sec=t1.sec-t2.sec;
    }
    if(t1.sec>=t2.sec)
```

```
    {
      t.sec=t1.sec-t2.sec;
    }

    if(t1.mi<t2.mi)
    {
      t1.mi=t1.mi+60;
      t1.hr=t1.hr-1;
      t.mi=t1.mi-t2.mi;
    }
    if(t1.mi>=t2.mi)
    {
      t.mi=t1.mi-t2.mi;
    }

    if(t1.hr<t2.hr)
    {
      t.hr=t1.hr-t2.hr;
    }
    if(t1.mi>=t2.mi)
    {
      t.hr=t1.hr-t2.hr;
    }
*/
    T1=t1.hr*3600+t1.mi*60+t1.sec;
    T2=t2.hr*3600+t2.mi*60+t2.sec;
    T=T1-T2;
```

```
    if(T1<T2)
        T=-T;

    if((T/60.0)>1)
    {
        t.mi=int(T/60.0);
        t.sec=float(((T/60.0)-int(T/60.0))*60.0);
        //cout<<float(((T/60.0)-int(T/60.0))*60.0)<<endl;
    }
    else
    {
        t.sec=T;
    }

    if((t.mi/60.0)>1)
    {
        t.hr=float(t.mi/60.0);
        t.mi=float(((t.mi/60.0)-int(t.mi/60.0))*60);
    }
    else
    {
        t.hr=0;
    }
    return t;
}

bool operator >(Time t1,Time t2)
{
```

```
  float ts1,ts2;
  ts1=(t1.hr*60*60)+(t1.mi*60)+t1.sec;
  ts2=(t2.hr*60*60)+(t2.mi*60)+t2.sec;

  if(ts1>ts2)
  {
     return true;
  }
  else
  {
     return false;
  }
}

int main()
{
   Time t1(10,10,4),t2(9,15,3),t3;
   bool result;

   t3=t1-t2;
   t3.show();

   result=t1>t2;
   cout<<result<<endl;



   return 0;
```

```
}
```

```
#include <iostream>

using namespace std;

class String
{
   int l;
   char strin[100];
public:
   void getdata()
   {
      cout<<"Enter number of character:"<<endl;
      cin>>l;

      cout<<"Enter character:"<<endl;
      cin>>strin;
      strin[l]='\0';
   }

   void showdata()
   {
```

```
        cout<<"Final string is: "<<endl;
        cout<<strin<<endl;
    }

    friend String operator +(String,String);
    friend bool operator ==(String,String);

};

String operator +(String s1,String s2)
{
    int i,j=0;
    String s;
    s.l=s1.l+s2.l;
    for(i=0;i<s1.l;i++)
    {
        s.strin[i]=s1.strin[i];
    }
    for(i+1;i<s.l;i++)
    {
        s.strin[i]=s2.strin[j];
        j++;
    }
    s.strin[i]='\n';
        return s;
}

bool operator ==(String s1, String s2)
```

```
{

    if(s1.l!=s2.l)
       return 0;

    for(int i=0;i<s1.l;i++)
    {
       if(int(s1.strin[i])!=int(s2.strin[i]))
       {
            return 0;

       }

    }

    return 1;


}

int main()
{
   String s1,s2,s3;

   s1.getdata();
   s2.getdata();

   s3=s1+s2;
```

```
    s3.showdata();

    bool flag;//by default this is 0
    flag=(s1==s2);
    cout<<"Result of comparision is: "<<flag<<endl;

    return 0;
}
```

```
#include <iostream>
#include <cstring>

using namespace std;

class String
{
    int l;
    char *strin;
public:

    ~String()
    {
        delete []strin;
```

```
    cout<<"deleted"<<endl;
}

void getdata()
{
    cout<<"Enter number of character:"<<endl;
    cin>>l;
    strin=new char[l+1];
    cout<<"Enter character:"<<endl;
    cin>>strin;//cin.getline not working
}

void showdata()
{
    cout<<"Final string is: ";
    cout<<strin<<endl;
}

String operator +(String s1)
{
    String s;
    s.l=l+s1.l;
    s.strin=new char[s.l+1];
    strcpy(s.strin,strin);
    strcat(s.strin,s1.strin);
    return s;
}
```

```
bool operator ==(String s1)
{

   if(s1.l!=l)
      return 0;

   for(int i=0;i<s1.l;i++)
   {
      if(int(s1.strin[i])!=int(strin[i]))
      {
            return 0;

      }

   }

   return 1;



}

void operator =(String s1)
{

l=s1.l;
strin =new char[l+1];
for(int i=0;i<s1.l;i++)
{
```

```
        strin[i]=s1.strin[i];
    }
        strin[l]='\0';
    }



};



int main()
{
    String s1,s2,s3,s4;

    s1.getdata();
    s2.getdata();

    s3=s1+s2;
    s4=s3;
    s3.showdata();
    if(s1==s2)
        cout<<"Two string are same!"<<endl;
    else
        cout<<"Two string are not same!"<<endl;

    return 0;
}
```

```
#include <iostream>

using namespace std;

class Time
{
   int hr,mi,sec;

public:
      friend istream & operator >>(istream &,Time &);
      friend ostream & operator <<(ostream &,Time);
};

istream & operator >> (istream &in ,Time &t1)
{
   cout<<"Enter time to be displayed:"<<endl;
   in>>t1.hr>>t1.mi>>t1.sec;

   return in;
}

ostream & operator << (ostream &out ,Time t1)
{
   cout<<"The time is:"<<endl;
```

```
    out<<t1.hr<<" hrs "<<t1.mi<<" min "<<t1.sec<<" secs";

    return out;
}

int main()
{
    Time t;
    cin>>t;
    cout<<t;
    return 0;
}
```

```
#include <iostream>

using namespace std;

class Matrix
{
    int m,n;
    int M[10][10];
```

```
public:
   void read()
   {
      cout<<"Enter number of row and column of matrix:"<<endl;
      cin>>m>>n;
      cout<<"Enter elements of matrix:"<<endl;
      for(int i=0;i<m;i++)
      {
         for (int j=0;j<n;j++)
         {
            cin>>M[i][j];
         }
      }
   }

   void display()
   {
      cout<<"Matrix is:"<<endl;
      for(int i=0;i<m;i++)
      {
         for (int j=0;j<n;j++)
         {
            cout<<M[i][j]<<"  ";
         }
         cout<<endl;
      }
   }
```

```
    friend Matrix operator +(Matrix,Matrix);
    friend bool is_valid (Matrix,Matrix);
};

Matrix operator +(Matrix m1,Matrix m2)
{
    Matrix m;
    m.m=m1.m;
    m.n=m1.n;
    for(int i=0;i<m.m;i++)
    {
        for(int j=0;j<m.n;j++)
        {
            m.M[i][j]=m1.M[i][j]+m2.M[i][j];
        }
    }

    return m;
}

bool is_valid(Matrix m1, Matrix m2)
{
    if((m1.m==m2.m)&&m1.n==m2.n)
        return true;
    else
        return false;
}
```

```
int main()
{
   Matrix m1,m2,m3;

   m1.read();
   m2.read();
   if (is_valid(m1,m2))
   {
      m3=m1+m2;
      m3.display();
   }
   else
      cout<<"Invalid matrices!!!";

   return 0;
}
```