```
// SPECIFICATION FILE for the time class          ( inherit1.h )
class  Time
{
public :

      void Set ( int  hours , int  minutes , int  seconds ) ;
      void Increment ( ) ;
      virtual void Write ( )  const ;
      Time ( int  initHrs, int  initMins,  int  initSecs ) ;   //  constructor
      Time ( ) ;                                              //  default constructor
      virtual ~Time ( ) ;
 private :

      int             hrs ;
      int             mins ;
      int             secs ;
} ;
//*****************************************************
//  IMPLEMENTATION FILE for the time class       ( inherit1i.cpp )
//   Implements the Time member functions.

#include <iostream>
using namespace std;
#include "inherit1.h"

// private data members
//           int hrs ;
//           int mins ;
//           int secs ;

void  Time::Set(int hours, int minutes, int seconds)
      {
            hrs = hours ;
            mins = minutes ;
            secs = seconds ;
      }


void  Time::Increment()
      {
            secs++ ;
            if (secs > 59)
                  {
                        secs = 0;
                        mins++;
                        if (mins > 59)
                              {
                                    mins = 0;
                                    hrs++;
                                    if (hrs > 23)
                                          hrs=0;
                              }
                  }
      }
```

```
void   Time :: Write ( )   const
     //   Postcondition:   Time has been output in form HH:MM:SS

      {     if  ( hrs < 10 )
                        cout << '0' ;
            cout  << hrs  <<  ':' ;
            if  ( mins < 10 )
                        cout << '0' ;
            cout  << mins  <<  ':' ;
            if  ( secs < 10 )
                        cout << '0' ;
            cout  << secs  ;
      }


Time :: Time (  ) : hrs(0), mins(0), secs(0)
{      // empty body
}

Time :: Time (   int   initHrs,  int   initMins, int   initSecs ):
               hrs (initHrs),
               mins (initMins),
               secs (initSecs)
{ // empty   body
}


 Time ::~Time ( ) {}

//*******************************************************
// SPECIFICATION   FILE for the ExtTime  class        ( inherit1e.h)

/*
 * inherit1e.h
 *
 *  Created on: Dec 14, 2016
 *      Author: jlebowitz
 */

#ifndef INHERIT1E_H_
#define INHERIT1E_H_
// needed to verify consistency between the derived and base classes
#include   "inherit1.h"
enum  ZoneType {EST, CST, MST, PST, EDT, CDT, MDT, PDT } ;
```

```
class  ExtTime  :  public  Time                // Time is the base class
{
public :
       void  Set ( int  hours,  int  minutes,  int   seconds ,
           ZoneType  timeZone ) ;

        void Write ( )  const ;

       ExtTime ( int  initHrs ,  int  initMins ,  int  initSecs ,
                           ZoneType  initZone ) ;        // constructor
       ExtTime ( ) ;        // default constructor
       virtual ~ExtTime ( ) ;      // default destructor

private :
       ZoneType  zone ;                        //  added data member
} ;
//*************
```

```
#endif
*///*****************************************************************
```

// implementation file for the ExtTime class (inherit1ei.cpp)

```
#include "inherit1e.h"
#include <iostream>
using namespace std;
#include <string>
using namespace std;


// additional private member of class:
//    ZoneType zone;
//
ExtTime :: ExtTime( /* in */    int                initHrs,
                              /* in */     int                initMins,
                              /* in */     int                initSecs,
                                    /* in */     ZoneType   initZone )

      :  Time (initHrs, initMins, initSecs)   // base constructor initializer
```

```
//  Precondition:  0 <= initHrs <= 23    &&    0 <= initMins <= 59
//                     0 <= initSecs <= 59   &&   initZone is assigned
//  Postcondition:
//           zone == initZone  && Time set by base class constructor
{
            zone  = initZone ;
}
//*********************************************************
 ExtTime :: ExtTime (  ) : Time()     // Note
//  Default  Constructor
//  Postcondition:

//              hrs == 0   &&   mins == 0   &&  secs == 0

//            (via  an  implicit call to base class default constructor )

//    &&  zone == EST
{
            zone  =  EST ;
}

//*************************************************************
void  ExtTime :: Set  ( /* in */     int                hours,
                                /* in */     int                minutes,
                                /* in */     int                seconds,
                                      /* in */     ZoneType   timeZone )

//  Precondition:  0 <= hours <= 23    &&    0 <= minutes <= 59
//                     0 <= seconds <= 59   &&   timeZone is assigned
//  Postcondition:
//           zone == timeZone  && Time set by base class function
{
                          Time :: Set (hours, minutes, seconds);  // calls base
constructor
            zone  = timeZone ;
}


//*************************************************************
void  ExtTime :: Write (   )   const

//  Postcondition:
//           Time has been output in form HH:MM:SS  ZZZ
//           where  ZZZ is the time zone abbreviation
{
            static  string  zoneString[8]  =
            {
                          "EST", "CST", "MST", "PST", "EDT", "CDT", "MDT", "PDT"
            } ;
           Time :: Write ( ) ;
                     cout  << ' ' <<  zoneString [zone]  << endl;
}

//*************************************************************************
```

```
ExtTime ::~ExtTime ( ) {}
```

// inherit1.cpp
// client for the Time and ExtTime classes

```
#include  "inherit1e.h"
#include <iostream>
using namespace std;

int main()
{
```

```
      #include  "inherit1e.h"
#include <iostream>
using namespace std;

int main()
 {

        Time        firstTime ( 3, 5,7);
        firstTime.Write( ) ;
        cout << endl;
        Time        secondTime;
        secondTime.Write( ) ;
        cout << endl;

        ExtTime    thisTime ( 8, 35, 0, PST ) ;
        thisTime.Write( ) ;
        ExtTime    thatTime ;
        thatTime.Write( ) ;

        firstTime.Set (10, 49, 23) ;
        firstTime.Write( ) ;
        cout << endl;

        thatTime.Set (7, 39, 25, CDT) ;
        thatTime.Write( ) ;

        firstTime.Increment ();
        firstTime.Write( );
        cout << endl;


     thatTime.Increment ( ) ;
     thatTime.Write ( ) ;
  }
```

output

```
03:05:07
00:00:00
08:35:00 PST
```

```
00:00:00 EST
10:49:23
07:39:25 CDT
10:49:24
07:39:26 CDT
```

```
//**********************************************************************
//inherit2.h header file for the PersonType class

#ifndef H_PersonType
#define H_PersonType

#include <string>

using namespace std;

class personType
{
public:
    void print() const;
         //Function to output the first name and last name
         //in the form firstName lastName

    void setName(string first, string last);
         //Function to set firstName and lastName according to
         //the parameters
         //Post: firstName = first; lastName = last;

    void getName(string& first, string& last);
         //Function to return firstName and lastName via the parameters
         //Post: first = firstName; last = lastName;

    personType(string first, string last);
         //Constructor with parameters
         //Set firstName and lastName according to the parameters
         //Post: firstName = first; lastName = last;

    personType();
         //Default constructor;
         //Intialize firstName and lastName to empty string
         //Post: firstName = ""; lastName = "";

private:
    string firstName; //store the first name
    string lastName;  //store the last name
};
#endif

//inherit2i.cpp implementation file for the PersonType class

#include <iostream>
```

```cpp
using namespace std;
#include <string>
#include "inherit2.h"

using namespace std;


void personType::print() const
{
	cout<<firstName<<" "<<lastName;
}

void personType::setName(string first, string last)
{
	firstName = first;
	lastName = last;
}

void personType::getName(string& first, string& last)
{
	first = firstName;
	last = lastName;
}

	//constructor with parameters
personType::personType(string first, string last)

{
	firstName = first;
	lastName = last;
}

personType::personType()   //default constructor
{
	firstName = "";
	lastName = "";
}
```

// header file for the partTimeEmployee

```cpp
#include "inherit2.h"

class partTimeEmployee: public personType
{
public:
    void print();
		//Function to output the first name, last name, and
		//the wages in the form:
		//firstName lastName wages are $$$$.$$

    double calculatePay();
		//Function to calculate and return the wages

    void setNameRateHours(string first, string last,
                          double rate, double hours);
```

```
            //Function to set the first name, last name, payRate,
            //and hoursWorked according to the parameters.
            //The parameters first and last are passed to the
            //base class. payRate = pay; hoursWorked = hours;

    partTimeEmployee(string first, string last,
                              double rate, double hours);
            //Constructor with parameters
            //Set the first name, last name, payRate, and
            //hoursWorked according to the parameters.
            //Parameters first and last are passed to the
            //base class. payRate = pay; hoursWorked = hours;

    partTimeEmployee();
            //Default constructor
            //Set the first name, last name, payRate, and
            //hoursWorked to the default values.
            //The first name and last name are initialized to an empty
            //string by the default constructor of the base class.
            //payRate = 0; hoursWorked = 0;

private:
    double payRate;     //store the pay rate
    double hoursWorked; //store the hours worked
};
```

//Implementation File partTimeEmployee class

```
#include <iostream>
#include "inherit2.h"
#include "partTimeEmployee.h"
using namespace std;

void partTimeEmployee::print()
{
      personType::print();      //print the name of the employee
      cout<<" wages are : "<<calculatePay()<<endl;
}

double partTimeEmployee::calculatePay()
{
      return (payRate * hoursWorked);
}

void partTimeEmployee::setNameRateHours(string first,
               string last, double rate, double hours)
{
      personType::setName(first,last);
payRate = rate;
      hoursWorked = hours;
}

partTimeEmployee::partTimeEmployee(string first, string last,
                  double rate, double hours)
```

```
        : personType(first, last) //constructor with parameters
{
        payRate = rate;
        hoursWorked = hours;
}

partTimeEmployee:: partTimeEmployee()   // default constructor
{
        payRate = 0;
        hoursWorked = 0;
}
```

//client for TimeEmployee

```
#include <iostream>

#include "inherit2.h"
#include "partTimeEmployee.h"

using namespace std;

int main()
{
        personType newPerson;
        partTimeEmployee newEmployee("John","Smith",7.50,56);
        partTimeEmployee employee;
        newEmployee.print();
        employee.setNameRateHours("Rachel", "Moore",9.75, 45);
        employee.print();

        return 0;
}
```

output

```
John Smith wages are : 420
Rachel Moore wages are : 438.75
```

//**********

// comp1.h (compostion)
//specification for the timecard class

```
#include  "inherit1.h"

        class TimeCard
        {
        public:
        void  Punch ( /* in */  int   hours, /* in */  int   minutes, /* in */  int   seconds ) ;
                                void  Print ( ) const ;
                                TimeCard (   long   idNum,
```

```
                                    int     initHrs,
                                    int     initMins,
                                    int     initSecs ) ;
                        TimeCard ( ) ;
            private:
                        long    id ;
                        Time    timeStamp ;
            } ;
//*************************************************************
// comp1i.cpp
// implementation file for the timecard class
#include "comp1.h"
#include <iostream>
using namespace std;

void TimeCard :: Print() const
      {
            cout << "ID: "<< id << " Time: " ; // invokes the Time method
            timeStamp.Write();
      }
//*********************************************************
TimeCard :: TimeCard  ( /* in */    long    idNum,
                                                /* in */    int      initHrs,
                                                /* in */    int      initMins,
                                               /* in */    int      initSecs )

       :  timeStamp (initHrs, initMins, initSecs)   // constructor initializer

{
                        id  =  idNum ;
}
//*********************************************************
TimeCard :: TimeCard()
{
                        id  =  0 ;
}

//*********************************************************
void TimeCard ::  Punch(int  hours, int minutes, int seconds)

      {
            timeStamp.Set(hours,minutes,seconds); // invokes the Time method
      }
//*************************************************************//


comp1.cpp
// client for the Timecard class
```

```
#include  "comp1.h"
#include <iostream>
using namespace std;

 int main()
 {
       TimeCard    thatTime ;            // default constructor called
       thatTime.Print( ) ;
        cout  <<  endl ;

       TimeCard   myTime (123,6,0,0) ;          //  constructor called
        myTime.Print( ) ;
        cout  <<  endl ;

        myTime.Punch(8,40,0 ) ;
        myTime.Print( ) ;
        cout  <<  endl ;
 }
```

output

```
ID: 0 Time: 00:00:00
ID: 123 Time: 06:00:00
ID: 123 Time: 08:40:00
```

```
//**********************************************************************
```

```
// comp2a.h (header file for the dateType class)
#ifndef date_H
#define date_H

class dateType
{
public:
   void setDate(int month, int day, int year);
               //Function to set the date
               //Data members dMonth, dDay, and dYear are set
               //according to the parameters
               //Post: dMonth = month; dDay = day;
               //                    dYear = year;

   void getDate(int& month, int& day, int& year);
               //Function to return the date
               //Post: month = dMonth; day = dDay;
               //                           year = dYear;

   void printDate() const;
```

```
            //Function to output the date in the form mm-dd-yyyy

   dateType(int month, int day, int year);
            //Constructor to set the date
            //Data members dMonth, dDay, and dYear are set
            //according to the parameters.
            //Post: dMonth = month; dDay = day;
            //            dYear = year;
            dateType();
            //Default constructor
            //Data members dMonth, dDay, and dYear are set to
            //the default values.
            //Post: dMonth = 1; dDay = 1; dYear = 1900;

private:
   int dMonth;              //variable to store the month
   int dDay;          //variable to store the day
   int dYear;         //variable to store the year
};
#endif
//comp2a.cpp (implementation file for the dateType class)
#include <iostream>
#include "comp2a.h"

using namespace std;

void dateType::setDate(int month, int day, int year)
{
        dMonth = month;
        dDay = day;
        dYear = year;
}

void dateType::getDate(int& month, int& day, int& year)
{
        month = dMonth;
        day = dDay;
        year = dYear;
```

```
}


void dateType::printDate() const
{
        cout<<dMonth<<"-"<<dDay<<"-"<<dYear;
}

        //constructor with parameter
dateType::dateType(int month, int day, int year)
{
        dMonth = month;
        dDay = day;
        dYear = year;
}

dateType::dateType()  //default parameter
{
        dMonth = 1;
        dDay = 1;
        dYear = 1900;
}
//comp2b.h (header for the person class)
#include <string>
using namespace std;

class personType
{
public:
   void print() const;
          //Function to output the first name and last name
          //in the form firstName lastName

   void setName(string first, string last);
          //Function to set firstName and lastName according to
          //the parameters
          //Post: firstName = first; lastName = last;
```

```
    void getName(string& first, string& last);
          //Function to return firstName and lastName via the parameters
          //Post: first = firstName; last = lastName;

    personType(string first, string last);
          //Constructor with parameters
          //Set firstName and lastName according to the parameters
          //Post: firstName = first; lastName = last;

    personType();
          //Default constructor;
          //Intialize firstName and lastName to empty string
          //Post: firstName = ""; lastName = "";

 private:
    string firstName; //store the first name
    string lastName;  //store the last name
};
```

//comp2b.cpp (implementation for the person class)

```
#include <iostream>
#include <string>
#include "comp2b.h"

using namespace std;


void personType::print() const
{
        cout<<firstName<<" "<<lastName;
}

void personType::setName(string first, string last)
{
        firstName = first;
        lastName = last;
}
```

```
void personType::getName(string& first, string& last)
{
        first = firstName;
        last = lastName;
}

        //constructor with parameters
personType::personType(string first, string last)

{
        firstName = first;
        lastName = last;
}

personType::personType()   //default constructor
{
        firstName = "";
        lastName = "";
}
// comp2c.h (header file for the personalInfo class)
#ifndef personalInfo_H
#define personalInfo_H

#include <string>
#include "comp2a.h"
#include "comp2b.h"

using namespace std;

class personalInfo
{
public:
   void setpersonalInfo(string first, string last, int month,
                        int day, int year, int ID);
        //Function to set the personal information.
        //Data members are set according to the parameters.
        //Post: firstName = first; lastName = last;
        //      dMonth = month; dDay = day; dYear = year;
```

```
        //    personID = ID;

    void printpersonalInfo () const;
        //Function to print personal information

    personalInfo(string first, string last, int month,
                 int day, int year, int ID);
        //Constructor with parameters.
        //Data members are set according to the parameters.
        //Post: firstName = first; lastName = last;
        //       dMonth = month; dDay = day; dYear = year;
        //  personID = ID;

    personalInfo();
        //Default constructor
        //Data members are set to the default values.

private:
    personType name;
    dateType bDay;
    int personID;
};
#endif
//comp2c.cpp (implementation for the personalInfo class)
#include <iostream>
#include <string>
#include "comp2c.h"

using namespace std;

void personalInfo::setpersonalInfo(string first, string last,
                                   int month, int day, int year, int ID)
{
        name.setName(first,last);
    bDay.setDate(month,day,year);
        personID = ID;
}
```

```
void personalInfo::printpersonalInfo() const
{
        name.print();
        cout<<"'s date of birth is ";
        bDay.printDate();
        cout<<endl;
        cout<<"and personal ID is "<<personID;
}

personalInfo::personalInfo(string first, string last, int month,
                int day, int year, int ID)
       : name(first,last), bDay(month,day,year)
{
        personID = ID;
}

personalInfo::personalInfo() //default constructor
{
        personID = 0;
}
//comp2.cpp (client)
#include <iostream>
#include "comp2c.h"

using namespace std;

int main()
{
        personalInfo newStudent("William", "Jordan", 8,24,1963,555238911);

        newStudent.printpersonalInfo();

        cout<<endl;

        return 0;
}
output
William Jordan's date of birth is 8-24-1963
```

and personal ID is 555238911

//**********

// multiple inheritance1

```
#include <iostream>

using namespace std;

// Base class Shape
class Shape
{
   public:
      void setWidth(int w)
      {
         width = w;
      }
      void setHeight(int h)
      {
         height = h;
      }
   protected:
      int width;
      int height;
};

// Base class PaintCost
class PaintCost
{
   public:
      int getCost(int area)
      {
         return area * 70;
      }
};

// Derived class
class Rectangle: public Shape, public PaintCost
{
   public:
      int getArea()
      {
         return (width * height);
      }
};

int main(void)
{
   Rectangle Rect;
   int area;

   Rect.setWidth(5);
   Rect.setHeight(7);
```

```
   area = Rect.getArea();

   // Print the area of the object.
   cout << "Total area: " << Rect.getArea() << endl;

   // Print the total cost of painting
   cout << "Total paint cost: $" << Rect.getCost(area) << endl;

   return 0;
}
```

Output

```
Total area: 35
Total paint cost: $2450
```

```
//*********************************************************************
// with private data members

#include <iostream>

using namespace std;

// Base class Shape
class Shape
{
   public:
      void setWidth(int w)
      {
         width = w;
      }
      void setHeight(int h)
      {
         height = h;
      }
      int getHeight() {
        return height;
      }
      int getWidth() {
        return width;
      }
   private:
      int width;
      int height;
};

// Base class PaintCost
class PaintCost
{
   public:
      int getCost(int area)
      {
         return area * 70;
      }
};

// Derived class
```

```
class Rectangle: public Shape, public PaintCost
{
   public:
      int getArea()
      {
         return (getWidth() * getHeight());
      }
};

int main(void)
{
   Rectangle Rect;
   int area;

   cout << " I am here";

   Rect.setWidth(5);
   Rect.setHeight(7);

   area = Rect.getArea();

   // Print the area of the object.
   cout << "Total area: " << Rect.getArea() << endl;

   // Print the total cost of painting
   cout << "Total paint cost: $" << Rect.getCost(area) << endl;

   return 0;
}
```

Output

```
Total area: 35
Total paint cost: $2450
```

// multiple inheritance2

```
#include <iostream>
using namespace std;
class Area
{
  public:
    float area_calc(float l,float b)
    {
        return l*b;
    }
};

class Perimeter
{
  public:
    float peri_calc(float l,float b)
    {
        return 2*(l+b);
    }
};
```

```
/* Rectangle class is derived from classes Area and Perimeter. */
class Rectangle : private Area, private Perimeter
{
    private:
        float length, width;
    public:
       Rectangle() : length(0.0), width(0.0) { }
       void get_data( )
       {
           cout<<"Enter length: ";
           cin>>length;
           cout<<"Enter width: ";
           cin>>width;
       }

       float area_calc()
       {
       /* Calls area_calc() of class Area and returns it. */

           return Area::area_calc(length,width);
       }

       float peri_calc()
       {
       /* Calls peri_calc() function of class Perimeter and returns it. */

           return Perimeter::peri_calc(length,width);
       }
};

int main()
{
    Rectangle r;
    r.get_data();
    cout<<"Area = "<<r.area_calc();
    cout<<"\nPerimeter = "<<r.peri_calc();
    return 0;
}
```

Output

```
Enter length: 44
Enter width: 55
Area = 2420
Perimeter = 198
```