```cpp
   void setGrossSales( double ); // set gross sales amount
   double getGrossSales() const; // return gross sales amount

   void setCommissionRate( double ); // set commission rate
   double getCommissionRate() const; // return commission rate

   void setBaseSalary( double ); // set base salary
   double getBaseSalary() const; // return base salary

   double earnings() const; // calculate earnings
   void print() const; // print BasePlusCommissionEmployee object
private:
   std::string firstName;
   std::string lastName;
   std::string socialSecurityNumber;
   double grossSales; // gross weekly sales
   double commissionRate; // commission percentage
   double baseSalary; // base salary
}; // end class BasePlusCommissionEmployee

#endif
```

Fig. 11.7 | BasePlusCommissionEmployee class header. (Part 2 of 2.)

```cpp
// Fig. 11.8: BasePlusCommissionEmployee.cpp
// Class BasePlusCommissionEmployee member-function definitions.
#include <iostream>
#include <stdexcept>
#include "BasePlusCommissionEmployee.h"
using namespace std;

// constructor
BasePlusCommissionEmployee::BasePlusCommissionEmployee(
   const string &first, const string &last, const string &ssn,
   double sales, double rate, double salary )
{
   firstName = first; // should validate
   lastName = last; // should validate
   socialSecurityNumber = ssn; // should validate
   setGrossSales( sales ); // validate and store gross sales
   setCommissionRate( rate ); // validate and store commission rate
   setBaseSalary( salary ); // validate and store base salary
} // end BasePlusCommissionEmployee constructor

```

Fig. 11.8 | BasePlusCommissionEmployee class represents an employee who receives a base salary in addition to a commission. (Part 1 of 6.)

```
// set first name
void BasePlusCommissionEmployee::setFirstName( const string &first )
{
   firstName = first; // should validate
} // end function setFirstName

// return first name
string BasePlusCommissionEmployee::getFirstName() const
{
   return firstName;
} // end function getFirstName

// set last name
void BasePlusCommissionEmployee::setLastName( const string &last )
{
   lastName = last; // should validate
} // end function setLastName

```

Fig. 11.8 | `BasePlusCommissionEmployee` class represents an employee who receives a base salary in addition to a commission. (Part 2 of 6.)

```cpp
// return last name
string BasePlusCommissionEmployee::getLastName() const
{
   return lastName;
} // end function getLastName

// set social security number
void BasePlusCommissionEmployee::setSocialSecurityNumber(
   const string &ssn )
{
   socialSecurityNumber = ssn; // should validate
} // end function setSocialSecurityNumber

// return social security number
string BasePlusCommissionEmployee::getSocialSecurityNumber() const
{
   return socialSecurityNumber;
} // end function getSocialSecurityNumber

```

Fig. 11.8 | `BasePlusCommissionEmployee` class represents an employee who receives a base salary in addition to a commission. (Part 3 of 6.)

```cpp
// set gross sales amount
void BasePlusCommissionEmployee::setGrossSales( double sales )
{
   if ( sales >= 0.0 )
      grossSales = sales;
   else
      throw invalid_argument( "Gross sales must be >= 0.0" );
} // end function setGrossSales

// return gross sales amount
double BasePlusCommissionEmployee::getGrossSales() const
{
   return grossSales;
} // end function getGrossSales

// set commission rate
void BasePlusCommissionEmployee::setCommissionRate( double rate )
{
   if ( rate > 0.0 && rate < 1.0 )
      commissionRate = rate;
   else
      throw invalid_argument( "Commission rate must be > 0.0 and < 1.0" );
} // end function setCommissionRate
```

Fig. 11.8 | `BasePlusCommissionEmployee` class represents an employee who receives a base salary in addition to a commission. (Part 4 of 6.)

```cpp

// return commission rate
double BasePlusCommissionEmployee::getCommissionRate() const
{
   return commissionRate;
} // end function getCommissionRate

// set base salary
void BasePlusCommissionEmployee::setBaseSalary( double salary )
{
   if ( salary >= 0.0 )
      baseSalary = salary;
   else
      throw invalid_argument( "Salary must be >= 0.0" );
} // end function setBaseSalary

// return base salary
double BasePlusCommissionEmployee::getBaseSalary() const
{
   return baseSalary;
} // end function getBaseSalary

```

Fig. 11.8 | `BasePlusCommissionEmployee` class represents an employee who receives a base salary in addition to a commission. (Part 5 of 6.)

```cpp
// calculate earnings
double BasePlusCommissionEmployee::earnings() const
{
   return baseSalary + ( commissionRate * grossSales );
} // end function earnings

// print BasePlusCommissionEmployee object
void BasePlusCommissionEmployee::print() const
{
   cout << "base-salaried commission employee: " << firstName << ' '
      << lastName << "\nsocial security number: " << socialSecurityNumber
      << "\ngross sales: " << grossSales
      << "\ncommission rate: " << commissionRate
      << "\nbase salary: " << baseSalary;
} // end function print
```

Fig. 11.8 | `BasePlusCommissionEmployee` class represents an employee who receives a base salary in addition to a commission. (Part 6 of 6.)

11.3.2 Creating a `BasePlusCommissionEmployee` Class Without Using Inheritance (cont.)

Defining Class `BasePlusCommissionEmployee`

- The `BasePlusCommissionEmployee` header (Fig. 11.7) specifies class `BasePlusCommissionEmployee`'s `public` services, which include the `BasePlusCommissionEmployee` constructor and member functions `earnings` and `print`.
- Lines 15–31 declare `public` *get* and *set* functions for the class's `private` data members `firstName`, `lastName`, `socialSecurityNumber`, `grossSales`, `commissionRate` *and* `baseSalary`.

11.3.2 Creating a `BasePlusCommissionEmployee` Class Without Using Inheritance (cont.)

- Note the similarity between this class and class `Commission-Employee` (Figs. 11.4–11.5)—in this example, we won't yet exploit that similarity.
- Class `BasePlusCommissionEmployee`'s `earnings` member function computes the earnings of a base-salaried commission employee.

Testing Class `BasePlusCommissionEmployee`

- Figure 11.9 tests class `BasePlusCommissionEmployee`.

```cpp
// Fig. 11.9: fig11_09.cpp
// BasePlusCommissionEmployee class test program.
#include <iostream>
#include <iomanip>
#include "BasePlusCommissionEmployee.h"
using namespace std;

int main()
{
   // instantiate BasePlusCommissionEmployee object
   BasePlusCommissionEmployee
      employee( "Bob", "Lewis", "333-33-3333", 5000, .04, 300 );

   // set floating-point output formatting
   cout << fixed << setprecision( 2 );

   // get commission employee data
   cout << "Employee information obtained by get functions: \n"
      << "\nFirst name is " << employee.getFirstName()
      << "\nLast name is " << employee.getLastName()
      << "\nSocial security number is "
      << employee.getSocialSecurityNumber()
      << "\nGross sales is " << employee.getGrossSales()
      << "\nCommission rate is " << employee.getCommissionRate()
      << "\nBase salary is " << employee.getBaseSalary() << endl;
```

Fig. 11.9 | BasePlusCommissionEmployee class test program. (Part 1 of 3.)

```cpp

   employee.setBaseSalary( 1000 ); // set base salary

   cout << "\nUpdated employee information output by print function: \n"
      << endl;
   employee.print(); // display the new employee information

   // display the employee's earnings
   cout << "\n\nEmployee's earnings: $" << employee.earnings() << endl;
} // end main
```

Fig. 11.9 | `BasePlusCommissionEmployee` class test program. (Part 2 of 3.)

```
Employee information obtained by get functions:

First name is Bob
Last name is Lewis
Social security number is 333-33-3333
Gross sales is 5000.00
Commission rate is 0.04
Base salary is 300.00

Updated employee information output by print function:

base-salaried commission employee: Bob Lewis
social security number: 333-33-3333
gross sales: 5000.00
commission rate: 0.04
base salary: 1000.00

Employee's earnings: $1200.00
```

Fig. 11.9 | `BasePlusCommissionEmployee` class test program. (Part 3 of 3.)

11.3.2 Creating a `BasePlusCommissionEmployee` Class Without Using Inheritance (cont.)

Exploring the Similarities Between Class `BasePlusCommissionEmployee` and Class `CommissionEmployee`

- Most of the code for class `BasePlusCommissionEmployee` (Figs. 11.7–11.8) is similar, if not identical, to the code for class `CommissionEmployee` (Figs. 11.4–11.5).
- In class `BasePlusCommissionEmployee`, `private` data members `firstName` and `lastName` and member functions `setFirstName`, `getFirstName`, `setLastName` and `getLastName` are identical to those of class `CommissionEmployee`.
- Both classes contain `private` data members `socialSecurityNumber`, `commissionRate` and `grossSales`, as well as *get* and *set* functions to manipulate these members.

11.3.2 Creating a `BasePlusCommissionEmployee` Class Without Using Inheritance (cont.)

- The `BasePlusCommissionEmployee` constructor is *almost* identical to that of class `CommissionEmployee`, except that `BasePlusCommissionEmployee`'s constructor also sets the `baseSalary`.
- The other additions to class `BasePlusCommissionEmployee` are `private` data member `baseSalary` *and* member functions `setBaseSalary` and `getBaseSalary`.
- Class `BasePlusCommissionEmployee`'s `print` member function is *nearly identical* to that of class `CommissionEmployee`, except that `BasePlusCommissionEmployee`'s `print` also outputs the value of data member `baseSalary`.

11.3.2 Creating a `BasePlusCommissionEmployee` Class Without Using Inheritance (cont.)

- We literally *copied* code from class `CommissionEmployee` and *pasted* it into class `BasePlusCommissionEmployee`, then modified class `BasePlusCommissionEmployee` to include a base salary and member functions that manipulate the base salary.
- This *copy-and-paste approach* is error prone and time consuming.
- Worse yet, it can spread many physical copies of the same code throughout a system, creating