Chapter Review

1. What is a class?
2. How does a class accomplish abstraction, encapsulation, and data hiding?
3. What is the relationship between an object and a class?
4. In what way, aside from being functions, are class function members different from
class data members?
5. Define a class to represent a bank account. Data members should include the
depositor's name, the account number (use a string), and the balance. Member functions
should allow the following:
n Creating an object and initializing it.
n Displaying the depositor's name, account number, and balance
n Depositing an amount of money given by an argument
n Withdrawing an amount of money given by an argument
Just show the class declaration, not the method implementations. (Programming
Exercise 1 provides you with an opportunity to write the implementation.)
Programming Exercises 559
6. When are class constructors called? When are class destructors called?
7. Provide code for a constructor for the bank account class from Chapter Review
Question 5.
8. What is a default constructor? What is the advantage of having one?
9. Modify the `Stock` class definition (the version in `stock20.h`) so that it has member
functions that return the values of the individual data members. Note:A member
that returns the company name should not provide a weapon for altering the array.
That is, it can't simply return a `string` reference. It could return a `const` reference.
10. What are `this` and `*this`?

Programming Exercises

1. Provide method definitions for the class described in Chapter Review Question 5
and write a short program that illustrates all the features.

2. Here is a rather simple class definition:

```
class Person {
private:
static const LIMIT = 25;
string lname; // Person's last name
char fname[LIMIT]; // Person's first name
public:
Person() {lname = ""; fname[0] = '\0'; } // #1
Person(const string & ln, const char * fn =
"Heyyou"); // #2
// the following methods display lname and fname
void Show() const; // firstname lastname format
void FormalShow() const; // lastname, firstname
format
};
```

(It uses both a `string` object and a character array so that you can compare how
the two forms are used.) Write a program that completes the implementation by
providing code for the undefined methods.The program in which you use the class
should also use the three possible constructor calls (no arguments, one argument,
and two arguments) and the two display methods. Here's an example that uses the
constructors and methods:

```
Person one; // use default constructor
Person two("Smythecraft"); // use #2 with one default
argument
Person three("Dimwiddy", "Sam"); // use #2, no
defaults
one.Show();
cout << endl;
one.FormalShow();
```

```
// etc. for two and three
```

6. Here's a class declaration:

```
class Move
{
private:
double x;
double y;
public:
Move(double a = 0, double b = 0); // sets x, y to a,
b
showmove() const; // shows current x, y values
Move add(const Move & m) const;
// this function adds x of m to x of invoking object
to get new x,
// adds y of m to y of invoking object to get new y,
creates a new
// move object initialized to new x, y values and
returns it
reset(double a = 0, double b = 0); // resets x,y to
a, b
};
```

Create member function definitions and a program that exercises the class.