## Contents

Guided Exercise 1.1: Setting up your application development environment 1

Part 1—Starting the Database Server 2

Part 2—Setting up a workspace in Developer Studio 4

Part 3—Setting up the Server project 7

Guided Exercise 1.1: Setting up your application development environment, Wrap-up 13

Try It 2.1: Defining classes 15

Part 1—Creating the Emp class 16

Part 2—Defining data members for the Emp class 17

Part 3—Defining methods for the Emp class 18

Part 4—Adding the include file for the ttEmployee temp-table 19

Part 5—Creating the Dept class 20

Part 6—Defining data members for the Dept class 21

Part 7—Defining a constructor and methods for the Dept class 22

Solution, Part 1—Creating the Emp class 23

Solution, Part 2—Defining data members for the Emp class 26

Solution, Part 3—Defining methods for the Emp class 30

Solution, Part 4—Adding the include file for the ttEmployee temp-table 35

Solution, Part 5—Creating the Dept class 37

Solution, Part 6—Defining data members for the Dept class 39

Solution, Part 7—Defining a constructor and methods for the Dept class 41

Try It 2.1: Defining classes, Wrap-up 45

Try It 2.2: Working with Classes 47

Part 1—Implementing the methods for the Emp class 48

Part 2—Implementing the methods for the Dept class 49

Solution, Part 1—Implementing the methods of the Emp class 50

Solution, Part 2—Implementing the methods for the Dept class 52

Try It 2.2: Working with classes, Wrap-up 54

Try It 2.3: Testing classes 55

Part 1—Setting up a Test Project 56

Part 2—Writing the test procedure for the Emp class 57

Part 3—Testing the Emp class 58

Part 4—Writing the test procedure for the Dept class 59

Part 5—Testing the Dept class 61

Solution, Part 1—Setting up a Test project 62

Solution, Part 2—Writing the test procedure for the Emp class 68

Solution, Part 3—Testing the Emp class 72

Solution, Part 4—Writing the test procedure for the Dept class 74

Solution, Part 5—Testing the Dept class 80

Try It 2.3: Testing classes, Wrap-up 81

Try It 3.1: Using inheritance 83

Part 1—Modify the Emp class to support its derived classes 84

Part 2—Creating the Manager class 85

Part 3—Defining a constructor and a data member for the Manager class 86

Part 4—Defining methods for the Manager class 87

Part 5—Implementing the methods for the Manager class 88

Part 6—Importing the TeamMember and Dept classes 89

Part 7—Testing the inheritance hierarchy 90

Solution, Part 1—Modify the Emp class to support its derived classes 85

Solution, Part 2—Creating the Manager class 91

Solution, Part 3—Defining a constructor and a data member for the Manager class 92

Solution, Part 4—Defining methods for the Manager class 95

Solution, Part 5—Implementing the methods of the Manager class 99

Solution, Part 6—Importing the TeamMember and Dept classes 101

Solution, Part 7—Testing the inheritance hierarchy 103

Try It 3.1: Using inheritance, Wrap-up 108

Try It 3.2: Using an interface class 109

Part 1—Creating the IBusiness Unit interface class 110

Part 2—Defining data members for the IBusiness Unit interface class 111

Part 3—Defining methods for the IBusiness Unit interface class 112

Part 4—Creating the Company class 113

Part 5—Implementing a constructor, a destructor, and methods for the Company class 114

Part 6—Importing the Franchise class 115

Part 7—Testing the classes 116

Solution, Part 1—Creating the IBusiness Unit interface class 117

Solution, Part 2—Defining data members for the IBusinessUnit interface class 119

Solution, Part 3—Defining methods for the IBusinessUnit interface class 121

Solution, Part 4—Creating the Company class 123

Solution, Part 5—Implementing a constructor, a destructor, and methods for the Company class 126

Solution, Part 6—Importing the Franchise class 128

Solution, Part 7—Testing the classes 129

Try It 3.2: Using an interface class, Wrap-up 132

Try It 3.3: Using a singleton and creating classes dynamically 133

Part 1—Importing the Corporation class and the ttBusinessUnit include file 134

Part 2—Defining static data members for the Corporation class 135

Part 3—Defining a static constructor for the Corporation class 136

Part 4—Adding code to the InitializeBusinessUnit() method to create instances dynamically 137

Part 5—Testing the Corporation class 138

Solution, Part 1—Importing the Corporation class and the ttBusinessUnit include file 139

Solution, Part 2—Defining static data members for the Corporation class 141

Solution, Part 3—Defining a static constructor for the Corporation class 143

Solution, Part 4—Adding code to the InitializeBusinessUnit() method to create instances dynamically 144

Solution, Part 5—Testing the Corporation class 145

Try It 3.3: Using a singleton and creating classes dynamically, Wrap-up 146

Try It 3.4: Using events 147

Part 1—Defining and publishing an event in the Manager class 148

Part 2—Modify the Dept class to subscribe to the event 149

Part 3—Testing the Dept class event 150

Solution, Part 1—Defining and publishing an event in the Manager class 151

Solution, Part 2—Modify the Dept class to subscribe to the event 154

Solution, Part 3—Testing the Dept class event 157

Try It 3.4: Using events, Wrap-up 158