**C# Programming: 3 – Course Map**

## Notes to the Instructor:

* Timings for this course account for a total of 100 minutes for breaks independent of number of days scheduled.
* Allow 5 minutes at end of first session for evaluations.
* Example projects are available for you to OPTIONALLY use during lectures and demos.
* All demonstrations where students should follow along with the instructor and do the steps are designated with an \*:
* Demo\*: The students should do the steps in tandem with the teacher.
* Demo: The teacher is showing a point and the students should watch.
* A double \*\* is designated for demonstrations where the instructor is provided a project along with a Microsoft Word document that highlights the focal points in the code in yellow and bold:
* Print the Microsoft Word document to have on hand as you are going through the demo.
* For each lab, have the students open the lab project specified for the lab. Strongly suggest that while the student is completing the lab, the result of the lab project execution should be showing in the Command Prompt on the screen for students to see the results (you may need to complete the lab yourself to accomplish this).
* Cover all points in the Content column for Lecture All and Demo All in the Activities column. It is acceptable to alternate between the two types of activities when covering different content points in the same row.
* All points in the Content column are to be defined, described or explained.
* There is flexibility to use the tools available, such as whiteboard, slides, and demonstrations so long as all the items in the Content column are covered in the allotted time.
* **Quizzes** are intended to be an all-class activity.
* All slides are inC\_\_Programming\_Level\_3.pptx file.

**Goal:** Students will learn how to:

* Locate drives, directories and files and determine their status.
* Store the state of object instances into binary files using the process of serialization.
* Apply interfaces using object-oriented programming principles.
* Implement events using delegates.
* Apply synchronization in multi-threaded applications.

**Textbook:** *Pro C# 5.0 and the .NET 4.5 Framework*, Sixth Edition, by Andrew Troelsen

**Class Setup:**

* All files referenced are contained in the course materials located in T:\C# Curriculae VS 2010 – READ ONLY\C# Programming Level 3 2010. The READ ONLY property of these files is set to on. After copying this folder to a different location, then reset the READ ONLY property of these files to off.
* Copy all folders and files to the shared drive for students to access EXCEPT for those folders that have “ – INSTRUCTOR ONLY” included in the name.
* Copy the skill check project and solution ONLY at the time specified in the course map when the particular skill check is assigned.

**C# Programming: 3**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Time** | **Media** | **Content** | **Learning Activity** | |
| 30 min. | FLIPCHRT | **Course Objective**  **Introduction**   * Student introductions * Course objectives and timing * Student preparation for each class session * Administrative details: prerequisites, book * Classroom and building details: restrooms, deli, bookstore, use of cell phones, food and beverages in class * Content and purpose of class materials * File locations * Certificate requirement * Solicit a volunteer for first session evaluations * Prerequisite paragraph from catalog | **Lecture (0 min.)**   * As students enter the classroom, tell them to sign the roster and to pick up the class syllabus.   **Discussion (15 min.)**   * Have students introduce themselves by giving their name, programming experience, and reason for taking the course.   **Lecture (10 min.)**   * **Referring to the syllabus:** * Briefly describe how the students should prepare for the next class session. Do this only at the beginning of the first session. * **Read the paragraph in the college catalog** about course prerequisites. * Briefly cover classroom and building details. * **Referring to the textbook:** * The textbook is used in levels 2-4 of the C# programming classes. * Explain that references to the textbook will be made throughout the course. * **Referring to files**, describe the locations from where to: * Retrieve class files. * Save individual files. * Execute files. * **Referring to class files**, describe the following: * Exercises folder and location of the Microsoft Word document that specifies the exercise requirements within each section. * Outside Class Study Slides folder. Emphasize the need to study the slides before studying the textbook. | |
| **Objective 1: Use the File class to display specific file information.** | | | | |
| 20 min. | BOOK06  **Ch. 20** | **Sub Objective 1a: Use the File class to retrieve information about a file.**  Part 1: Describe how to access the drives on the computer   * DriveInfo class: * Location in .NET Framework. * Purpose. * GetDrives method. * Properties: IsReady, AvailableFreeSpace, Name, DriveType | **Part 1 of Sub Objective 1a: Describe how to access the drives on the computer**  **Instructor Note:** Print all the Microsoft Word documents in the Highlighted Code In Demo folder for all demonstration projects in this objective. The highlighted portions indicate what code needs to be emphasized during the demonstration of that project.  **Lecture**   * Show slides 1 to 4 to present course Objective 1.   **Instructor Note:** 01\_DriveInfo complete project is available  **Lecture and Demo All\*\***  **Lab 01 (10 min.)**   * Have students open the Lab01 project and complete the TODO comments. * Explain that TODO comments are always listed in the TaskList window. | |
| 35 min. | BOOK06  **Ch. 20** | **Sub Objective 1a: Use the File class to retrieve information about a file.**  Part 2: Describe how to access directories and files   * Location in .NET Framework of DirectoryInfo and FileInfo classes. * DirectoryInfo class: * Methods: Get Directories, Get Files * Properties: FullName, Name * FileInfo class properties: Name, FullName, LastWriteTime, Length * Apply recursion to access all levels of directories and files. | **Part 2 of Sub Objective 1a: Describe how to access directories and files**  **Instructor Note:** 02\_DirectoryFileInfo complete project is available.  **Lecture and Demo All\*\***  **Lab 02 (15 min.)**   * Have students open the Lab02 project and complete the TODO comments. | |
| 20 min. | BOOK06  **Ch. 20** | **Sub Objective 1a: Use the File class to retrieve information about a file.**  Part 3: Describe how to locate, open and read a specific type of file   * DirectoryInfo class: GetFiles method * FileInfo class: OpenText method * StreamReader class members: * EndOfStream property. * Peek() Method. * ReadLine() Method. * “using” block: Purpose, syntax for opening and using a file. | **Part 3 of Sub Objective 1a: Describe how to locate, open and read a specific type of file**  **Instructor Note:** 03\_LocateSpecificFiles complete project is available.  **Lecture and Demo All\*\***  **Lab 03 (10 min.)**   * Have students open the Lab03 project and complete the TODO comments. |
| 10 min. | BOOK06  **Ch. 20** | **Sub Objective 1b: Implement serialization to store an object instance into a file.**  Part 1: Explain serialization | **Part 1 of Sub Objective 1b: Explain serialization**  **Lecture**   * Show slides 5 to 8 to explain serialization. | |
| 40 min. | BOOK06  **Ch. 20** | **Sub Objective 1b: Implement serialization to store an object instance into a file.**  Part 2: Describe how to serialize a single object   * Serializable attribute: Define, purpose, syntax and rules for applying to classes * BinaryFormatter class: * Location in .NET Framework. * Serialize method: Purpose, syntax and rules. | **Part 2 of Sub Objective 1b: Describe how to serialize a single object**  **Instructor Note:** 04\_SerializeSingleObjects complete project is available  **Lecture and Demo All\*\***  **Lab 04 (15 min.)**   * Have students open the Lab04 project and complete the TODO comments. | |
| 10 min. | BOOK06  **Ch. 20** | **Sub Objective 1b: Implement serialization to store an object instance into a file.**  Part 3: Describe how to serialize a collection of objects   * BinaryFormatter class: Serialize method | **Part 3 of Sub Objective 1b: Describe how to serialize a collection of objects**  **Instructor Note:** 05\_SerializeCollectionOfObjects complete project is available  **Lecture and Demo All\*\*** | |
| 10 min. | BOOK06  **Ch. 20** | **Sub Objective 1c: Implement deserialization to retrieve an object instance stored in a file.**  Part 1: Explain deserialization | **Part 1 of Sub Objective 1c: Explain deserialization**  **Lecture**   * Show slide 9 to explain deserialization. | |
| 15 min. | BOOK06  **Ch. 20** | **Sub Objective 1c: Implement deserialization to retrieve an object instance stored in a file.**  Part 2: Describe how to deserialize a single object   * File class: OpenRead method to open a stream file. * Deserialization implementation: * Must be implemented in same namespace as object graph was serialized. * Deserialize method of the BinaryFormatter: * Returns object type. * Object must be cast to the original type. | **Part 2 of Sub Objective 1c: Describe how to deserialize a single object**  **Instructor Note:** 06\_DeserializeSingleObjects complete project is available  **Lecture and Demo All\*\***  **Lab 05 (10 min.)**   * Have students open the Lab05 project and complete the TODO comments. | |
| 5 min. | BOOK06  **Ch. 20** | **Sub Objective 1c: Implement deserialization to retrieve an object instance stored in a file.**  Part 3: Describe how to deserialize a collection of objects   * BinaryFormatter class: Deserialize method | **Part 3 of Sub Objective 1c: Describe how to deserialize a collection of objects**  **Instructor Note:** 07\_DeserializeCollectionOfObjects complete project is available  **Lecture and Demo All\*\*** | |
| 45 min |  | **Skill Check A**  **Skill Check Lab A: Use the File class to display specific file information.**  The student will be given:   * A written description of the application requirements called Skill Check A Requirements. The requirements will involve: * Using the File class to retrieve specific information about files in a given folder. * Creating a recursive method to cycle through all the files of each subfolder in the given folder. * Opening a specific file type to read and display its contents. * A folder containing a number of subfolders and files of different types and different dates. | **Skill Check Lab A**  **Instructor Note:** Copy the SkillCheckA project to the Skill Check Projects folder on the S Drive, and then inform the students to copy that folder to their location.  Using the application requirements and the folder, the student will:   1. Create a new C# console application. 2. Complete the code to fulfill the requirements. 3. Test and debug the application until it meets all requirements. 4. Run the final test and leave the console window open. 5. Inform the instructor when they are done with the final test.   **Teacher Verify Step (15 seconds per student)**   * Look at the results on the screen to see if the correct answer and messages are showing.   **Instructor Note:** Copy the SkillCheckASolution project to the Skill Check Solutions folder on the S drive after the skill check is done. | |
| **Objective 2: Implement polymorphism using an interface.** | | | | |
| 15 min. | BOOK06  **Ch. 8** | **Sub Objective 2a: Create and implicitly implement multiple interfaces.**  Part 1: Describe the function of an interface | **Part 1 of Sub Objective 2a: Describe the function of an interface**  **Instructor Note:** Print the Microsoft Word documents in the Highlighted Code In Demo folder for all Demonstration projects in this objective. The highlighted portions indicate what code needs to be emphasized.  **Lecture (5 min.)**   * Show slide 10 to present course Objective 2. * Show slides 11 to 15 to describe the function of an interface.   **Instructor Note (10 min.):** When using the provided demonstration projects, instructors should open 00\_Interfaces project to explain its theme in preparation for the first four demonstrations in this objective. | |
| 20 min. | BOOK06  **Ch. 8** | **Sub Objective 2a: Create and implicitly implement multiple interfaces.**  Part 2: Create and implement an interface   * Create an interface: * Syntax and rules. * Members allowed. * keyword “virtual” must be applied when overriding is allowed. * Implement an interface: Syntax and rules, shortcut to implement methods of an interface | **Part 2 of Sub Objective 2a: Create and implement an interface**  **Instructor Note:** 01\_Interfaces complete project is available  **Lecture and Demo All\*\***  **Lab 06 (10 min.)**   * Have students open the Lab06 project and complete the TODO comments. | |
| 10 min. | BOOK06  **Ch. 8** | **Sub Objective 2a: Create and implicitly implement multiple interfaces.**  Part 3: Compare the use of interfaces vs. abstract classes | **Part 3 of Sub Objective 2a: Compare the use of interfaces vs. abstract classes**  **Lecture**   * Show slides 16 to 21 to compare interfaces and abstract classes. | |
| 25 min. | BOOK06  **Ch. 8** | **Sub Objective 2a: Create and implicitly implement multiple interfaces.**  Part 4: Implement multiple interfaces   * Purpose. * Syntax and rules. * How to access implicitly implemented interface methods: * Using object reference. * Casting object reference to interface type and using interface reference. | **Part 4 of Sub Objective 2a: Implement multiple interfaces**  **Lecture**   * Show slides 22 to 23 to describe the use of multiple interfaces.   **Instructor Note:** 02\_Interfaces complete project is available  **Lecture and Demo All\*\***  **Lab 07 (10 min.)**   * Have students open the Lab07 project and complete the TODO comments. | |
| 5 min. | BOOK06  **Ch. 8** | **Sub Objective 2a: Create and implicitly implement multiple interfaces.**  Part 5: Extend and combine interfaces | **Part 5 of Sub Objective 2a: Extend and combine interfaces**  **Lecture**   * Show slides 24 to 27 to describe the use of multiple interfaces. | |
| 10 min. | BOOK06  **Ch. 8** | **Sub Objective 2a: Create and implicitly implement multiple interfaces.**  Part 6: Explain the advantages of casting an object reference to an interface type | **Lecture**   * Show slides 28 to 29 to describe the use of multiple interfaces.   **Lab 08 (5 min.)**   * Have students open the Lab08 project and complete the TODO comments. | |
| 20 min. | BOOK06  **Not in book** | **Sub Objective 2a: Create and implicitly implement multiple interfaces.**  Part 7: Explain how to implement interfaces in structs | **Lecture**   * Show slides 30 to 32 to describe the use of multiple interfaces. | |
| 30 min. | BOOK06  **Ch. 8** | **Sub Objective 2b: Apply polymorphism by executing a method through an interface reference.**   * Describe how to apply polymorphism through reference to an interface type. | **Instructor Note:** 03\_Interfaces complete project is available  **Lecture and Demo All\*\***  **Lab 09 (10 min.)**   * Have students open the Lab09 project and complete the TODO comments. | |
| 5 min. | BOOK06  **Ch. 8** | **Sub Objective 2c: Create an explicit implementation of an interface.**  Part 1: Explain the rationale for using explicit interface implementation | **Lecture**   * Show slides 33 to 39 to describe the use of multiple interfaces. | |
| 40 min. | BOOK06  **Ch. 8** | **Sub Objective 2c: Create an explicit implementation of an interface.**  Part 2: Describe how to use members of an explicit interface implementation   * Syntax and rules for : * Explicitly implementing interfaces. * Accessing members of explicit implementation. | **Instructor Note:** 04\_Interfaces complete project is available  **Lecture and Demo All\*\***  **Lab 10 (15 min.)**   * Have students open the Lab10 project and complete the TODO comments. | |
| 35 min. | BOOK06  **Ch. 13: Pgs. 473-496** | **Sub Objective 2d: Implement the IDisposable and IComparable interfaces.**  Part 1: Implement the IDisposable interface   * Dispose method: * Purpose: Clean up memory. * Syntax and rules for implementing. * “using” block: * Only allowed to create instances of classes that implement the IDisposable interface. * Dispose method in class instance is automatically called when code in “using” block has finished execution. | **Instructor Note:** 05\_IDisposable complete project is available  **Lecture and Demo All\*\***  **Lab 11A (10 min.)**   * Have students open the Lab11A project and complete the TODO comments. | |
| 45 min. | BOOK06  **Ch. 13: Pgs. 473-496** | **Sub Objective 2d: Implement the IDisposable and IComparable interfaces.**  Part 2: Modify a class by adding a destructor   * Destructor (Finalize method): Purpose, syntax and rules for implementing, when executed * Process used by Garbage Collector to clear memory of unreferenced object instances: * Tracks addresses of instances containing Finalize method. * Removes instances from heap with no Finalize method. * Keeps instances with Finalize method in heap until method is done executing. * Garbage Collector class: SuppressFinalize method: Purpose and syntax, called by Dispose method. | **Instructor Note:** 06\_IDisposableAndDestructor complete project is available.  **Lecture and Demo All\*\***  **Lab 11B (10 min.)**   * Have students open the Lab11B project and complete the TODO comments. | |
| 45 min. | BOOK06  **Ch. 8** | **Sub Objective 2d: Implement the IDisposable and IComparable interfaces.**  Part 3: Implement the IComparable interface   * IComparable interface: * Purpose: * Dictates how one instance of a type compares to another instance. * Required by Sort method of Array class and .NET collection types. * CompareTo method syntax and return value. | **Part 3 of Sub Objective 2d: Implement the IComparable interface**  **Instructor Note:** 07\_IComparable complete project is available.  **Lecture and Demo All\*\***  **Lab 12 (10 min.)**   * Have students open the Lab12 project and complete the TODO comments.   **Quiz 01: Class Participation (15 min.)**   * Print and hand out slides 1 to 2 on Quizzes.ppt to each student. * Give students about 5 minutes to find bugs. * For each slide ask students for their answers | |
| Over-view  10 min.  Lab  30 min |  | **Skill Check B**  **Skill Check Lab B: Implement polymorphism using an interface.**  The student will be given:   * A written description of the application requirements called Skill Check B Requirements. The requirements will involve: * Creating an interface that specifies a Print method. * Revising the abstract Account class to remove the abstract Print method. * Revising each of the two account type classes to implement the Print method explicitly through the interface. * Revising the service collection class, called BankAccounts, to execute the Print method of account type classes using polymorphism through the interface. * Implementing the IDispose interface in the Bank class (client) to make sure the two outside flat files are closed. * A completed version of a C# console application. * A flat file, called SkillCheckBAccountData.txt, which contains two records. The two records should specify all the data needed by each of the two account type classes. * A document, called SkillCheckBAccountDataSpecs.docx, which contains the specifications of the SkillCheckBAccountData.txt flat file. * A flat file, called SkillCheckBAccountUpdate.txt, which contains records for updating each of the two account type classes as follows: * One record for non numeric data when numeric data is expected. * One record for data that creates a violation causing the new custom Exception class to be thrown. * Multiple records containing valid data causing no exceptions. * A document, called SkillCheckBAccountUpdateSpecs.docx, which contains the specifications of the SkillCheckBAccountUpdate.txt flat file. | **Skill Check Lab B**  **Instructor Note:** Copy the SkillCheckB project to the Skill Check Projects folder on the S Drive, and then inform the students to copy that folder to their location.  Using the application requirements, flat files, flat file specifications, and the C# console application, the student will:   1. Revise the given C# console application to meet the specified requirements. 2. Test and debug the application until it meets all requirements. 3. Run the final test and leave the console window open. 4. Inform the instructor when they are done with the final test.   **Teacher Verify Step (15 seconds per student)**   * Look at the results on the screen to see if the correct answer and messages are showing.   **Instructor Note:** Copy the SkillCheckBSolution project to the Skill Check Solutions folder on the S drive after the skill check is done. | |
| **Objective 3: Set up a class that will raise an event to the subscriber when a specific situation occurs, and set up the subscriber to employ a given action when the event occurs.** | | | | |
| 25 min. | BOOK06  **Ch. 10** | **Sub Objective 3a: Declare and use a delegate referencing existing and anonymous methods.**  Part 1: Describe delegates and when to use them | **Part 1 of Sub Objective 3a: Describe delegates and when to use them**  **Instructor Note:** Print all the Microsoft Word documents in the Highlighted Code In Demo folder for all Demonstration projects in this objective. The highlighted portions indicate what code needs to be emphasized.  **Lecture**   * Show slides 40 to 41 to present course Objective 3. * Show slides 42 to 52 to introduce delegates.   **Instructor Note:** When using the demonstration project, instructors must do the following:   * Open the 00\_DelegatesAndEvents project and explain its theme in preparation for the first seven demonstrations in this objective. * Show the students how to create a Class Diagram and have it printed for them to use while reviewing the code. | |
| 30 min. | BOOK06  **Ch. 10** | **Sub Objective 3a: Declare and use a delegate referencing existing and anonymous methods.**  Part 2: Describe how to use a delegate   * Syntax and rules for: * Creating a delegate type. * Declaring and creating a delegate instance: Delegate inference, creating an explicit instance | **Part 2 of Sub Objective 3a: Describe how to use a delegate**  **Instructor Note:** 01\_DelegatesAndEvents complete project is available.  **Lecture and Demo All\*\***  **Lab 13 (15 min.)**   * Have students open the Lab13 project and complete the TODO comments. | |
| 5 min. | BOOK06  **Ch. 10** | **Sub Objective 3a: Declare and use a delegate referencing existing and anonymous methods.**  Part 3: Describe anonymous methods and when to use them | **Lecture**   * Show slides 53 to 60 to describe anonymous methods and when to use them. | |
| 20 min. | BOOK06  **Ch. 10** | **Sub Objective 3a: Declare and use a delegate referencing existing and anonymous methods.**  Part 4: Describe how to use an anonymous method   * Implement an anonymous method. * Call an anonymous method. | **Instructor Note:** 02\_DelegatesAndEvents complete project is available.  **Lecture and Demo All\*\***  **Lab 14 (10 min.)**   * Have students open the Lab14 project and complete the TODO comments. | |
| 5 min. | BOOK06  **Ch. 10** | **Sub Objective 3b: Implement a callback using a delegate.**  Part 1: Explain the purpose of a callback method | **Lecture**   * Show slide 61 to explain the purpose of a callback. | |
| 55 min. | BOOK06  **Not in book** | **Sub Objective 3b: Implement a callback using a delegate.**  Part 2: Describe how to setup a callback   * Describe the steps for setting up and using a callback method contained in a delegate | **Instructor Note:** 03\_DelegatesAndEvents complete project is available.  **Lecture and Demo All\*\***  **Lab 15 (20 min.)**   * Have students open the Lab15 project and complete the TODO comments. | |
| 10 min. | BOOK06  **Ch. 10** | **Sub Objective 3c: Implement a multicast delegate containing references to methods that return a value.**  Part 1: Describe a multicast delegate and how to use it | **Lecture**   * Show slides 62 to 67 to describe multicasting. | |
| 50 min. | BOOK06  **Ch. 10** | **Sub Objective 3c: Implement a multicast delegate containing references to methods that return a value.**  Part 2: Describe how to use a multicast delegate that does NOT return a value   * Multicast delegate: * How to instantiate with methods that do not return a value. * How to invoke. * Explain that the delegate type contains a collection of one or more delegate instances. | **Instructor Note:** 04\_DelegatesAndEvents complete project is available.  **Lecture and Demo All\*\***  **Lab 16 (15 min.)**   * Have students open the Lab16 project and complete the TODO comments. | |
| 20 min. | BOOK06  **Ch. 10** | **Sub Objective 3c: Implement a multicast delegate containing references to methods that return a value.**  Part 3: Describe how to use a multicast delegate that returns a value   * Create delegate instances with methods that return a value. * GetInvocationList method of delegate instance: purpose, return value * Issues when GetInvocationList method is not used to invoke one delegate at a time: * Only return value from last delegate will be captured. * If exception occurs on one of the delegates, remaining delegates are not invoked. | **Instructor Note:** 05\_DelegatesAndEvents complete project is available.  **Lecture and Demo All\*\***  **Lab 17 (10 min.)**   * Have students open the Lab17 project and complete the TODO comments. | |
| 15 min. | BOOK06  **Not in book** | **Sub Objective 3d: Apply covariance and contravariance when using a delegate.**  Part 1: Explain delegate covariance and contravariance | **Lecture**   * Show slides 68 to 69 to explain delegate covariance and contravariance. | |
| 65 min. | BOOK06  **Not in book** | **Sub Objective 3d: Apply covariance and contravariance when using a delegate.**  Part 2: Describe how to use delegate covariance and contravariance | **Instructor Note:** 11\_CovarianceAndContravarianceDemo complete project is available.  **Lecture and Demo All\*\***  **Lab 18 (20 min.)**   * Have students open the Lab18 project and complete the TODO comments. | |
| 20 min. | BOOK06  **Ch. 10** | **Sub Objective 3e: Declare and use an event that contains a reference to a custom EventArgs class.**  Part 1: Describe events and how to use them   * Describe events and how to use them. (Slides) * Demonstrate use of events in a C# Windows application: (Demo) * Explain the following about the Click event handler for a Button control: * Code in application that subscribes to the event. * Button control is the publisher. | **Part 1 of Sub Objective 3e: Describe events and how to use them**  **Lecture**   * Show slides 70 to 77 to describe events and how to use them.   **Instructor Note:** 12\_WindowsApplication1 complete project is available.  **Demo (5 min.)**   * Open a C# Windows application that has at least one Button control on the form. * Show the code for the Click event handler. * Show the code where the application subscribes to the delegate of the Click event. | |
| 35 min. | BOOK06  **Ch. 10** | **Sub Objective 3e: Declare and use an event that contains a reference to a custom EventArgs class.**  Part 2: Implement events   * Steps for publishing an event. * Steps for subscribing to an event. * Raise an event. | **Part 2 of Sub Objective 3e: Implement events**  **Instructor Note:** 09\_DelegatesAndEvents complete project is available.  **Lecture and Demo All\*\***  **Lab 19 (15 min.)**   * Have students open the Lab19 project and complete the TODO comments. | |
| 50 min. | BOOK06  **Ch. 10** | **Sub Objective 3e: Declare and use an event that contains a reference to a custom EventArgs class.**  Part 3: Describe how to use a custom EventArgs class   * Implement a custom EventArgs class. * Pass an instance of a custom EventArgs class to an event handler. * Access property values in an event handler. | **Part 3 of Sub Objective 3e: Describe how to use a custom EventArgs class**  **Instructor Note:** 10\_DelegatesAndEvents complete project is available.  **Lecture and Demo All\*\***  **Lab 20 (15 min.)**   * Have students open the Lab20 project and complete the TODO comments.   **Instructor Note:**   * Recommend to the students to study the 08\_DelegatesAndEvents demonstration, which demonstrates the use of a delegate within a test harness.   **Quiz 02: Class Participation (15 min.)**   * Print and hand out slides 3 to 5 on Quizzes.ppt to each student. * Give the students about 5 minutes to find bugs. * For each slide: * Ask students for their answers. * Explain the answers as needed. | |
| 25 min |  | **Skill Check C**  **Skill Check Lab C: Set up a class that will raise an event to the subscriber when a specific situation occurs, and set up the subscriber to employ a given action when the event occurs.**  The student will be given:   * A written description of the application requirements called Skill Check C Requirements. The requirements will involve: * Creating a new EventArgs type class to contain the interest rate. * Revising the service collection class, called BankAccounts, as follows: * Subscribe to the Bank class in the constructor. * Create an event handler to call the CalculateInterest method through the interface of each account instance in the collection. * Revising the Bank class (client) to: * Declare the delegate for the event. * Declare the event using the delegate. * Raise the event to all subscribers using the custom EventArgs class. This code should be in the loop of the new decimal type array mentioned below. * Revising the interface implementation of the CalculateInterest method in the CD and Checking classes. * A completed version of the C# console application from Skill Check B with revisions to the Bank class (client) as follows: * A new instance of a single dimensional array of decimal data type with two elements containing interest rates. * Functionality to loop through the new decimal array with a sleep method of about 250 milliseconds. * A flat file, called SkillCheckCAccountData.txt, which contains two records. The two records should specify all the data needed by each of the two account type classes. * A document, called SkillCheckCAccountDataSpecs.docx, which contains the specifications of the SkillCheckCAccountData.txt flat file. * A flat file, called SkillCheckCAccountUpdate.txt, which contains records for updating each of the two account type classes as follows: * One record for non-numeric data when numeric data is expected. * One record for data that creates a violation causing the new custom Exception class to be thrown. * Multiple records containing valid data causing no exceptions. * A document, called SkillCheckCAccountUpdateSpecs.docx, which contains the specifications of the SkillCheckCAccountUpdate.txt flat file. | **Skill Check Lab C**  **Instructor Note:** Copy the SkillCheckC project to the Skill Check Projects folder on the S Drive, and then inform the students to copy that folder to their location.  Using the application requirements, flat files, flat file specifications, and the C# console application, the student will:   1. Revise the given C# console application to meet the specified requirements. 2. Test and debug the application until it meets all requirements. 3. Run the final test and leave the console window open. 4. Inform the instructor when they are done with the final test.   **Teacher Verify Step (15 seconds per student)**   * Look at the results on the screen to see if the correct answer and messages are showing.   **Instructor Note:** Copy the SkillCheckCSolution project to the Skill Check Solutions folder on the S drive after the skill check is done. | |
| **Objective 4: Set up a multi-threaded application that employs synchronization of common resources and ensures that the application will end only after all threads have completed processing.** | | | | |
| 30 min. | BOOK06  **Ch. 19: Pgs. 697-731** | **Sub Objective 4a: Create and use a delegate to invoke a method asynchronously.**  Part 1: Describe multi-threaded processes   * Multi-threaded processes. (Slides 80 to 91) * Path of execution when a method is called synchronously using a delegate. (Demo) | **Instructor Note:** Print all the Microsoft Word documents in the Highlighted Code In Demo folder for all Demonstration projects in this objective.  **Lecture (15 min.)**   * Show slides 78 to 79 to present course Objective 4. * Show slides 80 to 91 to describe multi-threaded processes.   **Instructor Note (5 min.):** When using the provided demonstrations projects, instructors must open the 02\_AsyncDelegates project and explain its theme in preparation for the demonstrations in this objective.  **Instructor Note:** 02\_AsyncDelegates complete project is available.   * **Demo (10 min.)** show how a method executes synchronously using a delegate. | |
| 5 min. | BOOK06  **Ch. 19: Pgs. 697-731** | **Sub Objective 4a: Create and use a delegate to invoke a method asynchronously.**  Part 2: Describe asynchronous programming | **Lecture**   * Show slides 92 to 95 to introduce asynchronous programming. | |
| 25 min. | BOOK06  **Ch. 19: Pgs. 697-731** | **Sub Objective 4a: Create and use a delegate to invoke a method asynchronously.**  Part 3: Implement asynchronous programming   * BeginInvoke method of delegate instance: * Purpose. * Syntax and rules for implementing asynchronous calls. * Variation of signature depending on signature of method being called. * Returns IAsyncResult type. * EndInvoke method of delegate instance: * Purpose. * Variation of signature depending on signature of method that was called. * Capture return value of method referenced within delegate. | **Instructor Note:** 03\_AsyncDelegates complete project is available.  **Lecture and Demo All\*\***  **Lab 21 (10 min.)**   * Have students open the Lab21 project and complete the TODO comments. | |
| 15 min. | BOOK06  **Ch. 19: Pgs. 697-731** | **Sub Objective 4a: Create and use a delegate to invoke a method asynchronously.**  Part 4: Implement polling of asynchronous calls   * IsCompleted property of the IAsyncResult object. | **Instructor Note:** 04\_AsyncDelegates complete project is available.  **Lecture and Demo All\*\***  **Lab 22 (10 min.)**   * Have students open the Lab22 project and complete the TODO comments. | |
| 20 min. | BOOK06  **Ch. 19: Pgs. 697-731** | **Sub Objective 4a: Create and use a delegate to invoke a method asynchronously.**  Part 5: Implement the WaitOne method for asynchronous calls   * IAsyncResult object: * AsyncWaitHandle.WaitOne method: * Purpose. * Syntax and rules for using. * TimeOut argument: Purpose. * Boolean return value: Purpose. | **Instructor Note:** 05\_AsyncDelegates complete project is available.  **Lecture and Demo All\*\***  **Lab 23 (5 min.)**   * Have students open the Lab23 project and complete the TODO comments. | |
| 45 min. | BOOK06  **Ch. 19: Pgs. 697-731** | **Sub Objective 4a: Create and use a delegate to invoke a method asynchronously.**  Part 6: Implement a callback for asynchronous calls   * BeginInvoke method of delegate containing asynchronous method: * Argument for AsyncCallback delegate instance: Method signature allowed * Callback method: * Automatically called when asynchronous method is done executing. * Executed on secondary thread. * Code to retrieve return value from asynchronous method. | **Instructor Note:** 06\_AsyncDelegates complete project is available.  **Lecture and Demo All\*\***  **Instructor Note:** The lab below takes extra time for students to implement a callback for asynchronous calls.  **Lab 24 (25 min.)**   * Have students open the Lab24 project and complete the TODO comments. | |
| 25 min. | BOOK06  **Ch. 19: Pgs. 697-731** | **Sub Objective 4a: Create and use a delegate to invoke a method asynchronously.**  Part 7: Describe how to pass data to the callback method   * BeginInvoke method(): Data object argument. * ASyncState property of IAsyncResult argument in callback method: Object data type, cast value before using | **Instructor Note:** 07\_AsyncDelegates complete project is available.  **Lecture and Demo All\*\***  **Lab 25 (10 min.)**   * Have students open the Lab25 project and complete the TODO comments. | |
| 10 min. | BOOK06  **Ch. 19: Pgs. 697-731** | **Sub Objective 4b: Implement a secondary thread to invoke a method asynchronously.**  Part 1: Describe the Thread Class | **Lecture**   * Show slides 96 to 98 to describe the Thread Class. | |
| 40 min. | BOOK06  **Ch. 19: Pgs. 697-731** | **Sub Objective 4b: Implement a secondary thread to invoke a method asynchronously.**  Part 2: Implement multi-threading using the ThreadStart delegate   * ThreadStart delegate: * Purpose. * Syntax for creating an instance. * Only parameterless methods can be referenced. * Create and start a secondary thread using instance of ThreadStart delegate. * Technique for passing values to/from asynchronous method. | **Instructor Note:** 08\_Threading complete project is available.  **Lecture and Demo All\*\***  **Lab 26 (10 min.)**   * Have students open the Lab26 project and complete the TODO comments. | |
| 25 min. | BOOK06  **Ch. 19: Pgs. 697-731** | **Sub Objective 4b: Implement a secondary thread to invoke a method asynchronously.**  Part 3: Implement multi-threading using the ParameterizedThreadStart delegate   * ParameterizedThreadStart delegate: * Purpose. * Syntax for creating an instance. * Only methods with a single parameter of object data type can be referenced. * Create and start a secondary thread using instance of ParameterizedThreadStart delegate. | **Instructor Note:** 09\_Threading complete project is available.  **Lecture and Demo All\*\***  **Lab 27 (15 min.)**   * Have students open the Lab27 project and complete the TODO comments. | |
| 40 min. | BOOK06  **Ch. 19: Pgs. 697-731** | **Sub Objective 4b: Implement a secondary thread to invoke a method asynchronously.**  Part 4: Describe thread properties   * Thread class properties: Name, ManagedThreadID, CurrentThread, IsAlive | **Instructor Note:** 10\_Threading complete project is available.  **Lecture and Demo All\*\***  **Lab 28 (10 min.)**   * Have students open the Lab28 project and complete the TODO comments. | |
| 20 min. | BOOK06  **Ch. 19: Pgs. 697-731** | **Sub Objective 4c: Join a thread to stall the main thread until a secondary thread is done executing.**  Part 1: Explain thread lifetimes | **Lecture**   * Show slides 99 to 107 to explain thread lifetimes. | |
| 30 min. | BOOK06  **Ch. 19: Pgs. 697-731** | **Sub Objective 4c: Join a thread to stall the main thread until a secondary thread is done executing.**  Part 2: Describe how to interrupt a paused thread   * Thread class methods: * Sleep static method: * Purpose. * Syntax and rules for pausing a thread. * Best practice: Sleep method needs to be wrapped in a separate try/catch block to catch the ThreadInterruptedException. * Interrupt instance method: * Purpose. * How executed in asynchronous method. * Throws the ThreadInterruptedException. | **Instructor Note:** 11\_Threading complete project is available.  **Lecture and Demo All\*\***  **Lab 29 (10 min.)**   * Have students open the Lab29 project and complete the TODO comments. | |
| 25 min. | BOOK06  **Ch. 19: Pgs. 697-731** | **Sub Objective 4c: Join a thread to stall the main thread until a secondary thread is done executing.**  Part 3: Describe how to join a secondary thread to the main thread   * Join method of thread object: Purpose, syntax, TimeOut argument, Bool return value | **Instructor Note:** 12\_Threading complete project is available.  **Lecture and Demo All\*\***  **Lab 30 (10 min.)**   * Have students open the Lab30 project and complete the TODO comments. | |
| 25 min. | BOOK06  **Ch. 19: Pgs. 697-731** | **Sub Objective 4c: Join a thread to stall the main thread until a secondary thread is done executing.**  Part 4: Describe how to join multiple secondary threads to the main thread | **Instructor Note:** 13\_Threading complete project is available.  **Lecture and Demo All\*\***  **Lab 31 (10 min.)**   * Have students open the Lab31 project and complete the TODO comments. | |
| 5 min. | BOOK06  **Ch. 19: Pgs. 697-731** | **Sub Objective 4c: Join a thread to stall the main thread until a secondary thread is done executing.**  Part 5: Explain the thread Priority property | **Lecture**   * Show slides 108 to 111 to explain the thread Priority property. | |
| 15 min. | BOOK06  **Ch. 19: Pgs. 697-731** | **Sub Objective 4c: Join a thread to stall the main thread until a secondary thread is done executing.**  Part 6: Describe how a thread’s priority can affect its execution   * Priority property of thread object: Syntax and rules | **Instructor Note:** 14\_Threading complete project is available.  **Lecture and Demo All\*\***   * Run the same project multiple times with different thread priorities so the students can see the difference.   **Lab 32 (5 min.)**   * Have students open the Lab32 project and complete the TODO comments. | |
| 20 min. | BOOK06  **Ch. 19: Pgs. 697-731** | **Sub Objective 4c: Join a thread to stall the main thread until a secondary thread is done executing.**  Part 7: Describe how to create a thread using a lambda expression   * Lambda expression: * Purpose. * Syntax variations and rules. * Lambda operator. * Same CIL code whether or not lambda expression is used. | **Instructor Note:** 14B\_LambdaThreading complete project is available.  **Lecture and Demo All\*\***   * Display the CIL code for all four ways a thread instance is created.   **Lab 33 (5 min.)**   * Have students open the Lab33 project and complete the TODO comments. | |
| 5 min. | BOOK06  **Not in book** | **Sub Objective 4c: Join a thread to stall the main thread until a secondary thread is done executing.**  Part 8: Describe exception handling in a multi-threaded application | **Lecture**   * Show slide 112 to explain exception handling in a multi-threaded application. | |
| 15 min. | BOOK06  **Not in book** | **Sub Objective 4c: Join a thread to stall the main thread until a secondary thread is done executing.**  Part 9: Describe how to process exceptions on secondary threads   * Implementation of exception handling in all secondary entry methods. | **Instructor Note:** 14C\_Threading complete project is available.  **Lecture and Demo All\*\***  **Lab 34 (10 min.)**   * Have students open the Lab34 project and complete the TODO comments. | |
| 10 min. | BOOK06  **Ch. 19: Pgs. 697-731** | **Sub Objective 4d: Synchronize a common resource.**  Part 1: Explain the basics of using synchronization in a multi-threaded application | **Lecture**   * Show slides 113 to 122 to explain the basics of using synchronization in a multi-threaded application. | |
| 30 min. | BOOK06  **Ch. 19: Pgs. 697-731** | **Sub Objective 4d: Synchronize a common resource.**  Part 2: Describe how to implement synchronization using the Monitor class   * Singleton object: Purpose in multi-threaded applications, how to create * Monitor class: * Purpose. * How to implement using the Enter and Exit methods. * Best practice:  1. Surround code between Enter and Exit methods in a try block. 2. Include Exit method in finally block to ensure that it always runs. | **Instructor Note:** 15\_Threading complete project is available.  **Lecture and Demo All\*\***  **Lab 35 (10 min.)**   * Have students open the Lab35 project and complete the TODO comments. | |
| 15 min. | BOOK06  **Ch. 19: Pgs. 697-731** | **Sub Objective 4d: Synchronize a common resource.**  Part 3: Describe how to implement synchronization using the “lock” keyword   * “lock” keyword: * Purpose. * CIL code generated. * Advantage over using Monitor class for synchronization. * Best practice: Surround “lock” block with Try/Catch to catch an exception that occurs in block when necessary. | **Instructor Note:** 16\_Threading complete project is available.  **Lecture and Demo All\*\***  **Demo**   * Display the CIL code where the “lock” keyword is being used in the source code to show the students that the compiler replaces the “lock” keyword with the implementation of the Monitor class.   **Lab 36 (5 min.)**   * Have students open the Lab36 project and complete the TODO comments. | |
| 15 min. | BOOK06  **Not in book** | **Sub Objective 4d: Synchronize a common resource.**  Part 4: Describe how to prepare for an infinite loop in a critical code path   * TryEnter vs. Enter methods of the Monitor class. * Use Monitor class to synchronize a critical code path that could include an infinite loop. | **Instructor Note:** 16B\_Threading complete project is available.  **Lecture and Demo All\*\*** | |
| 50 min. | BOOK06  **Ch. 19: Pgs. 697-731** | **Sub Objective 4d: Synchronize a common resource.**  Part 5: Describe how to implement the AutoResetEvent class for synchronization   * AutoResetEvent class: * Purpose. * Force main thread to wait until secondary threads are completed: * Create an instance of AutoResetEvent type without being in a signaled state. * Explain the purpose, thread used to execute, and syntax for the WaitOne and Set methods: | **Instructor Note (15 min.):** 16C\_Threading complete project is available. To prepare students for the demonstration project:   * Create, print, and hand out a Class View Diagram of the project. * Print and hand out the 16C\_Threading\_Demo\_Overview.docx file Explain the project using these documents prior to and during the demonstration of the project.   **Lecture and Demo All\*\***  **Lab 37 (10 min.)**   * Have students open the Lab37 project and complete the TODO comments. | |
| 20 min. | BOOK06  **Ch. 19: Pgs. 697-731** | **Sub Objective 4d: Synchronize a common resource.**  Part 6: Describe how to implement the Interlocked class for synchronization   * Interlocked class: * Purpose. * Syntax and use for the following methods: * Increment * Add * Decrement * CompareExchange | **Instructor Note:** 16D\_Threading complete project is available.  **Lecture All**  **Instructor Note:** When students are done with the lab below, ask if they saw the wakeup message appear more than once in the Console window. Why is it best to place the increment of the variable inside the lock block?  **Lab 38 (10 min.)**   * Have students open the Lab38 project and complete the TODO comments. | |
| 5 min. | BOOK06  **Ch. 19: Pgs. 697-731** | **Sub Objective 4e: Activate background threads from the thread pool.**  Part 1: Describe the basics of using a thread pool | **Lecture**   * Show slides 123 to 127 to describe the basics of using a thread pool. | |
| 30 min. | BOOK06  **Ch. 19: Pgs. 697-731** | **Sub Objective 4e: Activate background threads from the thread pool.**  Part 2: Describe how to implement the thread pool   * ThreadPool class: QueueUserWorkItem method: Purpose, syntax | **Instructor Note:** 17\_Threading complete project is available.  **Lecture and Demo All\*\***  **Lab 39 (15 min.)**   * Have students open the Lab39 project and complete the TODO comments. | |
| 5 min. | BOOK06  **Not in book** | **Sub Objective 4e: Activate background threads from the thread pool.**  Part 3: Explain how to debug a multi-threaded application | **Lecture**   * Show slides 128 to 131 to explain how to debug a multi-threaded application. | |
| 25 min. | BOOK06  **Not in book** | **Sub Objective 4e: Activate background threads from the thread pool.**  Part 4: Describe how to use Visual Studio tools to debug a multi-threaded application   * Threads window: Purpose, how to display and use * Callstack window: Purpose, how to display particular thread * Disassembly window: Purpose, view the source code | **Instructor Note:** 16\_Threading complete project is available.  **Lecture and Demo All\*\***  **Quiz 04: Class Participation (15 min.)**   * Print and hand out slides 6 to 7 on Quizzes.ppt to each student. * Give the students about 5 minutes to find bugs. * For each slide Ask students for their answers | |
| 30 min |  | **Skill Check D**  **Skill Check Lab D:** Set up a multi-threaded application that employs synchronization of common resources and ensures that the application will end only after all threads have completed processing.  The student will be given:   * A written description of the application requirements called Skill Check D Requirements. The requirements will involve: * Revising the event handler in the service collection class, called BankAccounts, as follows: * Create a separate thread to call the CalculateInterest of each account in the collection asynchronously. * Add functionality to ensure that all threads have completed processing before the event handler is allowed to end. * Revising the CalculateAccountInterest method in the BankAccounts class to synchronize processing and prevent problems from arising when using a common resource in multi-threaded applications. * A completed version of the C# console application from Skill Check C. * A flat file, called SkillCheckDAccountData.txt, which contains two records. The two records should specify all the data needed by each of the two account type classes. * A document, called SkillCheckDAccountDataSpecs.docx, which contains the specifications of the SkillCheckDAccountData.txt flat file. * A flat file, called SkillCheckDAccountUpdate.txt, which contains records for updating each of the two account type classes as follows: * One record for non-numeric data when numeric data is expected. * One record for data that creates a violation causing the new custom Exception class to be thrown. * Multiple records containing valid data causing no exceptions. * A document, called SkillCheckDAccountUpdateSpecs.docx, which contains the specifications of the SkillCheckDAccountUpdate.txt flat file. | **Skill Check Lab D**  **Instructor Note:** Copy the SkillCheckD project to the Skill Check Projects folder on the S Drive, and then inform the students to copy that folder to their location.  Using the application requirements, flat files, flat file specifications, and the C# console application, the student will:   1. Revise the given C# console application to meet the specified requirements. 2. Test and debug the application until it meets all requirements. 3. Run the final test and leave the console window open. 4. Inform the instructor when they are done with the final test.   **Teacher Verify Step (15 seconds per student)**   * Look at the results on the screen to see if the correct answer and messages are showing.   **Instructor Note:** Copy the SkillCheckDSolution project to the Skill Check Solutions folder on the S drive after the skill check is done. | |
| 5 min. |  | **Conclusion**   * Provide a class evaluation. | **Conclusion**  **Lecture (5 min.)**   * Explain how to provide a class evaluation. | |