# Lab: Core I/O Programming

**Scenario**

Reading and writing files are common tasks in programming. Applications need to persist data in some form, such as a database or local files, to store user configuration data. This lab focuses on using .NET classes to read and write text files on the local system.

**Objectives**

At the end of this lab, you will be able to:

• Write content to the Console window.

• Accept input from the user in the Console window.

• Write text files.

• Read text files.

**Lab Setup**

Start the 10975A-SEA-DEV virtual machine if it is not already running, and log on as **Student** with the password **Pa$$w0rd**.

## Exercise 1: Reading and Writing with the Console

**Scenario**

Sometimes applications do not contain a GUI with buttons and menus, but rather they exist as command line or console-based applications. Users interact with these applications by reading text that is displayed on the screen and by entering text or commands to have the application perform a task.

The main tasks for this exercise are as follows:

1. Write output to the console.

2. Read information from the console.

### Task 1: Write output to the console

1. Go to the AllFiles\Mod09\Labfiles\Starter folder, and then open the Console\_IO\_CS project.

2. Write output to the Console window to prompt the user for deciding to create a Student class or an Instructor class.

3. Use the following format to output the information to the console:

Do you want to create a:

S – Student

I – Instructor

Q - Quit

### Task 2: Read information from the console

• Using the project from the previous task, read the response from the user by using the **Console.Read()** method. Store the value in a **char** variable.

**Results**: After completing this exercise, you would have written content to the Console window and read input from the keyboard.

## Exercise 2: Reading and Writing Files

**Scenario**

This exercise will focus on persisting data about a student or an instructor to a file on the local computer. It makes use of the **StreamReader** and **StreamWriter** classes and writing text files.

The main tasks for this exercise are as follows:

1. Write text files.

2. Read text files.

### Task 1: Write text files

**Note:** This exercise builds on the previous one by using the console I/O code. After a user selects the appropriate option, the application creates either a student or instructor object, prompts for input, and then stores that information in the object.

1. Go to the AllFiles\Mod09\Labfiles\Starter folder, and then open the File\_IO\_CS project.

2. Write the information from the student or the instructor object that is created in the application code, to a file for persistence.

3. Read the file, populate a new object with the information, and then display the results on the Console window.

### Task 2: Read text files

1. Go to the AllFiles\Mod09\Labfiles\Starter folder, and then open the File\_IO\_CS project.

2. Read from a previously created text file for a student or instructor, and output the contents to the Console window.

3. Create a **ReadFile(string filename)** method that opens the file, and then reads the entire content.

4. Call the **ReadFile(string filename)** method at the end of the **WriteContents()** method.

**Results**: At the end of this exercise you will have persisted student and instructor data to a text file.