MOC 10975 A Introduction to .NET Programming

Course Summary

Description

In this 5-day course, students will learn the basics of computer programming through the use of Microsoft Visual Studio 2013 and either the Visual C# or Visual Basic programming languages. The course assumes no prior programming experience and introduces the concepts needed to progress to the intermediate courses on programming, such as 20483B: Programming in C#.

The focus will be on core programming concepts such as computer storage, data types, decision structures, and repetition by using loops. The course also covers an introduction to object-oriented programming covering classes, encapsulation, inheritance, and polymorphism. Coverage is also included around exception handling, application security, performance, and memory management.

Objectives

At the end of this course, students will be able to:

- Explain core programming fundamentals such as computer storage and processing.
- Explain computer number systems such as binary.
- Create and use variables and constants in programs.
- Explain how to create and use functions in a program.
- Create and use decisions structures in a computer program.
- Create and use repetition (loops) in a computer program.
- Explain pseudocode and its role in programming.
- Explain the basic computer data structures such as arrays, lists, stacks, and queues.
- Implement object-oriented programming concepts.
- Create and use classes in a computer program.
- Implement encapsulation, inheritance, and polymorphism.
- Describe the base class library (BCL) in the .NET Framework.
- Explain the application security concepts.
- Implement simple I/O in a computer program.
- Identify application errors and explain how to debug an application and handle errors.
- Identify the performance considerations for applications.

Topics

- Introduction to Core Programming Concepts
- Core Programming Language Concepts
- Program Flow
- Algorithms and Data Structures
- Error Handling and Debugging
- Introduction to Object-Oriented Programming
- More Object-Oriented Programming
- Introduction to Application Security
- Core I/O Programming
- Application Performance and Memory Management

Audience

This course is intended for anyone who is new to software development and wants, or needs, to gain an understanding of programming fundamentals and object-oriented programming concepts. They will typically be high school students, post-secondary school students, or career changers, with no prior programming experience. They might want to gain an understanding of the core programming fundamentals before moving on to more advanced courses such as 20483B: Programming in C#.

MOC 10975 A Introduction to .NET Programming

Course Summary (cont'd)

Prerequisites

Before attending this course, students must have:

- Ability to use computers to start programs, open and save files, navigate application menus and interfaces
- Ability to understand logical concepts such as comparisons
- Understand number theory
- Ability to create, understand, and follow structured directions or step-by-step procedures
- Ability to understand and apply abstract concepts to concrete examples

Duration

Five days

MOC 10975 A Introduction to .NET Programming

Course Outline

I. Introduction to Core Programming Concepts

This module provides background and foundational information on how computers process information, discusses the different types of applications that a programmer might be creating, and then provides information on how code is compiled and interpreted by a computer

A. Computer Data Storage and Processing
B. Application Types
C. Application Life-Cycle
D. Code Compilation

Lab: Thinking Like a Computer

- Creating Step-by-Step Directions for a Morning Routine

II. Core Programming Language Concepts

This module covers programming language syntax and the importance of using good syntax and following the syntax rules for the chosen language. This module also discusses the core data types and how to store these data types in computer memory by using variables and constants.

A. Syntax
B. Data Types
C. Variables and Constants

Lab: Working with Data Types•Selecting Data Types

- Declaring and Using Variables for Numeric Types
- Declaring and Using Variables for Textual Data Types
- Working with Boolean Variables
- Declaring and Using Constants

III. Program Flow

This module covers how code is executed in a computer program, such as top to bottom, in structured programming and branching in code execution. The module teaches these concepts through the use of functions, decision structures, and looping constructs.

A. Introduction to Structured Programming Concepts
B. Introduction to Branching
C. Using Functions
D. Using Decision Structures
E. Introducing Repetition

Lab: Creating Functions, Decisions, and Looping

- Implementing Functions
- Implementing Decisions in Code
- Implementing Repetition Structures

IV. Algorithms and Data Structures

This module introduces the concept of an algorithm by examining a daily routine such as a morning routine for getting up and going to work, outlining all the steps required including the decisions to be made as the routine progresses. The module also discusses how to translate these set of steps into pseudo code for evaluation of the algorithm that will be translated into actual code

A. Understand How to Write Pseudo Code
B. lgorithm Examples
C. Introduction to Data Structures

Lab: Working with Algorithms and Data Structures

- Working with Pseudo Code
- Creating Data Structures

V. Error Handling and Debugging

This module helps students understand that errors are a part of programming and they must understand how to anticipate errors, handle those errors in code, and present a good user experience with a program. This module introduces structured exception handling as the mechanism to deal with errors

A. Introduction to Program Errors
B. Introduction to Structured Error Handling
C. Introduction to Debugging in Visual Studio

Lab: Implementing Debugging and Error Handling

- Create Structured Exception Handlers
- Using the Visual Studio Debugger

VI. Introduction to Object-Oriented Programming

This module covers an introduction to the concepts related to object-oriented programming (OOP). The content has been split across two modules with this module focusing on basic OOP concepts that will provide sufficient knowledge to understand complex data structures starting with structs and then moving onto classes. This module helps the students gain an understanding of how to encapsulate data and related functionality within a class

A. Introduction to Complex Structures
B. Introduction to Structs
C. Introduction to Classes
D. Introducing Encapsulation

Lab: Implementing Complex Data Structures

- Creating structs
- Creating Classes

MOC 10975 A Introduction to .NET Programming

Course Outline (cont'd)

VII. More Object-Oriented Programming
This module teaches students about inheritance and polymorphism in classes and function overloading. Function overloading and polymorphism can go hand-in-hand as often times when you inherit from a class, you want to override or change the existing behavior to suit the needs of you class.
The module also provides an introduction to the base class library in the .NET Framework so that students can start to think about the existence of functionality in other class files and how they can search the .NET Framework to find this functionality and take advantage of it.

A. Introduction to Inheritance
B. Introduction to Polymorphism
C. Introduction to the .NET Framework and the Base Class Library

Lab: Implementing Inheritance

- Creating a Base Class
- Inheriting a Base Class

Lab: Implementing Polymorphism

- Implementing Polymorphism by Overriding a Function
- Implementing Polymorphism by Overloading

VIII. Introduction to Application Security
This module helps students think about security in their applications. This module introduces the concepts of authentication for users and also introduces the concept of permissions for the code that is running on a computer. It explains that operating systems might prevent certain aspects of the program from executing, such as saving a file to a directory to which the user running the app might not have permission to write. The module briefly covers code signing and why programmers might want to consider using code signing

A. Authentication and Authorization
B. Code Permissions on Computers
C. Introducing Code Signing

IX. Core I/O Programming
This module introduces some core input/output (I/O) concepts that programmers will use while creating applications. Starting with console I/O, this module introduces input and output to the Console window. The module also talks about reading and writing files, which is an important concept to know because applications work with the files on the disk systems on computers.

A. Lessons
B. Using Console I/O
C. Using File I/O

Lab: Core I/O Programming

- Reading and Writing with the Console
- Reading and Writing Files

X. Application Performance and Memory Management
This module enables students understand that memory on a computer is a finite resource. It talks about how good application design and good coding discipline with memory conservation and memory management will help programmers learn to develop applications that users will like. This is because these applications will be fast, responsive, and do not negatively impact other applications

A. Value Types vs Reference Types
B. Converting Types
C. The Garbage Collector

Lab: Using Value Types and Reference Types
Converting Types