SE020CS - *C# Debugging*
Duration: 2 days; / 14 hours; Instructor-led/ remote online training
Time: 9.00am – 5.00pm
Break: 10.15am – 10.30am /3.15pm – 3.30pm
Lunch: 1.00pm – 2.00pm

WHAT YOU WILL LEARN
The Visual Studio debugger helps you observe the run-time behavior of your program and find problems. The debugger works with all Visual Studio programming languages and their associated libraries. With the debugger, you can break execution of your program to examine your code, examine and edit variables, view registers, see the instructions created from your source code, and view the memory space used by your application.

This course is aim to equip students with debugging in C# application development.

AUDIENCE
C# developers.

PREREQUISITES
Before attending this course, students must be able to demonstrate the following skills:
- Basic knowledge and skill in C# language
- Basic knowledge and experience in using Microsoft Visual Studio.NET 2015 or Higher
- Experience in multithread and parallel application development are not essential, but having them will added advantage

The course materials, lectures, and lab exercises are in English. To benefit fully from the instruction, students need an understanding of the English language and completion of the prerequisites.

METHODOLOGY
This program will be conducted with interactive lectures, PowerPoint presentation, discussions and practical exercise

COURSE OBJECTIVES
Upon completion of this course, you will
- know what debugging is about
- perform debugging in fixing the logical and runtime errors
- be able to use the debugger effectively
- know how to debug multithread and parallel applications

COURSE OUTLINES

Module 1: Errors
- Fault, Error, and Failure
- Different Types of Error
- What is debugging?

Module 2: Getting Started with the Debugger
- Debug a Basic C# Project
- Inspect Variables
- Stepping Into and Over Function Calls
- Step Out

Module 3: Navigating through Code with the Debugger
- Start debugging
- Step into code, line by line
- Step through code, skipping functions
- Run to a specific location or function
- Move the pointer to change the execution flow
- Step into non-user code
- Step into properties and operators in managed code

Module 4: Using Breakpoints
- Setting a function breakpoint in source code
- Setting Other Kinds of Breakpoints
- Setting a Breakpoint in the Call Stack Window
- Setting a Breakpoint in the Disassembly Window
- Managing Breakpoints

Module 4: Advanced Breakpoints
- Breakpoint conditions
- Using Object IDs in Breakpoint Conditions
- Hit Count
- Filter
- Breakpoint Actions and Tracepoints
- Breakpoint labels
- Export and Import Breakpoints
- Troubleshoot breakpoints

Module 5: Debugger Windows
- Autos and Locals Windows
- Watch and QuickWatch Windows
- Disassembly Window

- Call Stack Window
- Registers Window
- Modules Window
- Memory Window

Module 6: Managing Exceptions with the Debugger

- Managing Exceptions with the Exception Settings Window
- Setting the debugger to continue on user-unhandled exceptions
- Continuing Execution After an Exception
- How to Examine System Code After an Exception?
- How to use Native Run-Time Checks?

Module 7: Attach to Running Processes with the Visual Studio Debugger

- Attach to a running process on the local machine
- Attach to a process on a remote computer
- Common debugging scenarios
- Troubleshoot attach errors

Module 8: Debug Multithreaded Applications

- Debug Threads and Processes
- Debug Multiple Processes
- How to use the Threads Window
- How to switch to Another Thread While Debugging
- How to flag and Unflag Threads
- How to set a Thread Name in Native Code
- How to set a Thread Name in Managed Code
- Tips for Debugging Threads in Native Code

Module 9: Parallel Applications Debugging

- Using the Tasks Window
- Using the Parallel Stacks Window
- How to use the Parallel Watch Window
- How to use the GPU Threads Window

Module 10: Others

- Just My Code
- Specify Symbol (.pdb) and Source Files in the Visual Studio Debugger
- Debug 64-Bit Applications
- Debugger Security