Advanced C# Programming in .NET

Assemblies
- What are assemblies?
- Modules
- Assemblies versus regular .DLLs
- Assemblies for reuse, versioning, deployment and security
- Assembly contents: Metadata, Resources, code and manifest
- Private and shared assemblies
- Creating and using assembly DLLs
- Assemblies and namespaces

Shared Assemblies and Versioning
- Shared assemblies
- Global Assembly Cache (GAC)
- Side by side versions and cultures
- Strong names
- Public and private key encryption
- Signing assemblies
- Installing assemblies in the GAC
- Delay signing
- Versioning
- Other assembly attributes

Disposal and Garbage Collection(GC)
- IDisposable, Dispose and Close
- Finalisers
- Automatic garbage collection
- GC internals
- Memory leaks

Design by Contract
- Background (Eiffel Programming language)
- Supplier and client: rights and responsibilities
- What is reliable and correct software?
- Correctness and Hoare triples
- Preconditions, postconditions and assertions
- Imperative and declarative statements

Code Contracts in .NET
- Overview of Code Contract
- The binary rewriter
- The Contract class
- Implementing contract by design
- Contracts on interfaces and abstract classes
- Dealing with contract failure

Delegates and Events
- Loose coupling using interfaces: *Strategy pattern*
- Delegates: safe function pointers
- Loose coupling using delegates: *Strategy pattern*
- Multicast delegates
- Events
- Publisher-Subscriber idiom
- Model-View / Observer pattern
- Defining and raising events
- Create event handlers and subscribers
- Anonymous methods versus Lambda functions
- Custom event storage

Reflection
- What is reflection?
- Metadata, data about data
- Reflection API: *Assembly, Module, Type* and *MemberInfo* classes
- Dynamic object creation
- Dynamic method invocation
- Dynamic assembly loading
- Dynamic programming
- Code emission
- Applications of reflection

Attributes
- What are attributes?
- Some intrinsic (build-in) attributes
- Creating custom attributes
- Reading attributes

Generics
- Traditional .NET object data structures
- Concrete type wrapper classes
- Generic .NET datastructures
- Collection initializers
- Creating generic classes
- Templates versus generics
- Default values
- Multiple generic types
- Generic type alias & var variables
- Generic derivation- and constructor constraints
- Generic methods
- Generic delegates and events

Advanced Generics
- Generics and reflection
- Generics and serialization
- Integrating Object Oriented Programming & Generic Programming
- Covariance & contravariance
- Strategy pattern with generics

Multi-Paradigm Programming in C#

- Object-oriented (OOP), generic (GP) and functional (FP) models
- Criteria for choosing a given model
- Delegates versus interfaces versus virtual/abstract methods
- Designing component-based systems in .NET

Processes and Threads

- Starting external processes
- Redirecting standard IO
- *Thread* class and *ThreadStart* and *ParameterizedThreadStart* delegates
- Thread life cycle
- Controlling thread execution
- Joining threads
- Thread synchronisation
- Synchronising collections
- Thread notification: *Wait*, *Pulse*, *PulseAll*
- Producer-Consumer pattern

An Introduction to Task Parallel Library (TPL)

- Tasks and futures
- The *Parallel* class
- Parallel Invoke, For and Foreach Task Manager

Networking

- URIs, URLs, IP addresses and DNS
- Simple networking with *WebRequest* & *WebResponse*
- Sockets
- User Datagram Protocol (UDP)
- Sending and receiving datagrams
- Transmission Control Protocol (TCP)
- *NetworkStream*
- Servers (*TcpListener*) and clients (*TcpClient*)

Dynamic Programming

- Dynamic Language Runtime (DLR)
- Dynamic member overload resolution
- Simplifying the Visitor pattern; multiple dispatch
- Dynamic objects
- Interoperability with dynamic languages
- Dynamic programming versus Reflection
- Large System Design in C#

System decomposition and application domains

- Single domain, multidomain and distributed applications
- Using multiple application domains
- Domains and threads
- Sharing data between domains

Advanced C# Language Features

- Object and Collection Initialisers
- Auto-Implemented Properties
- Implicit Typed Variables
- Anonymous Types
- Lambda Expressions
- Extension Methods
- Tuples

Introduction to LINQ

- LINQ query expressions
- Query operators
- Ordering, Subsets, Single element, Aggregation, Quantifying and Set Operators
- Sub queries
- Grouping and Joining

Interoperability with Legacy Code

- Using legacy DLLs
- DLL Callback functions
- Runtime Callable Wrapper
- Primary Interop Assemblies
- Using .NET components in non .NET software
- COM Callable Wrapper
- Registering .NET components for COM
- Early and late binding

Mixing C# with C++ using C++/CLI

- What is C++/CLI
- C# application using native C++
- Exposing C++ class to .NET using C++/CLI wrapper
- Native C++ application using C#
- Using a C# GUI in C++
- Interfacing to Boost C++ Libraries