ELE 503

Advanced Computer Programming and Statistics

Week#6: Introduction to C# Programming

By Kingsley E. Erhabor

Week 06.

Introduction to C# Programming

Introduction to C# Programming

Principles and Methodologies

Understand C# Syntax, Data Types, and Control Structures

Grasp the fundamental syntax rules of C#.

Write Simple C# Programs to Solve Engineering Problems

Apply programming concepts to real-world engineering scenarios

Explore the .NET Framework and Its Relevance to Engineering Applications

Comprehend the architecture and components of the .NET Framework

Q&A

Closing Take away

Efosa's Introduction

Engineer | Programmer | Innovator

Technical Authority

Shell Nigeria

Subject Mater Expert (EMEA) for Process Automation & Control (PACO)-Subsea control systems and Subsea Distribution

Containerization 85 %

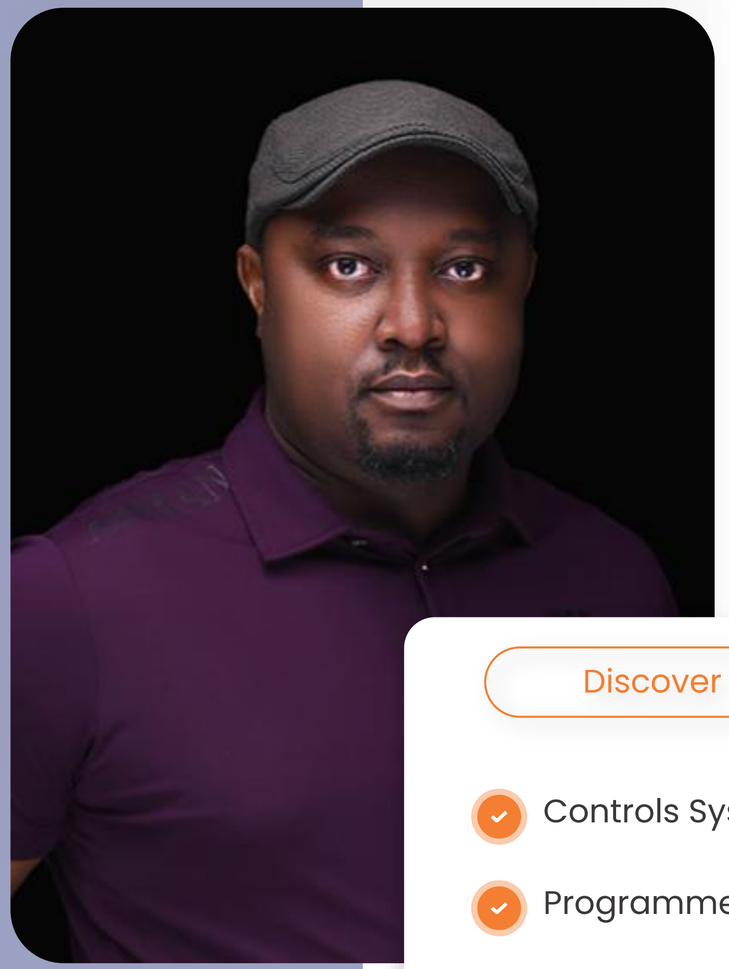
AI & ML 80 %

Renewable Energy 90 %

Innovator, VC

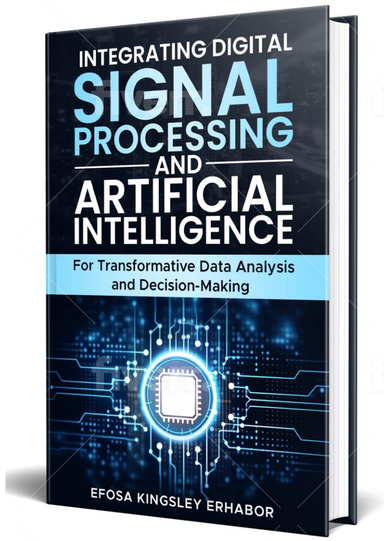
Katharos Technologies

Linux, Devops, AI and Software SME, Innovator and enterprenur



Discover

- Controls System
- Programmer
- Subsea Engineer
- AI and ML



X or Twitter

Linkedin

Part 1:

Introduction to C# Programming

Introduction to C#

ELE 503: Advanced Computer Programming and Statistics

- **What is C#?**
  - A modern, object-oriented programming language developed by Microsoft.
  - Part of the .NET ecosystem.
  - Designed for building a wide range of applications, from desktop to web to mobile.

- **Key Features:**
  - **Object-Oriented:** Supports encapsulation, inheritance, and polymorphism.
  - **Type-Safe:** Prevents type errors by enforcing strict type rules.
  - **Modern Syntax:** Clean and expressive, facilitating easier code maintenance.
  - **Versatile:** Suitable for developing various applications in engineering and beyond

C# History and Significance

ELE 503: Advanced Computer Programming and Statistics

- **Development Timeline:**
  - **2000:** C# was introduced by Microsoft as part of its .NET initiative.
  - **2002:** First version of the C# compiler released with .NET Framework 1.0.
  - **Ongoing:** Continuous updates with new features and enhancements.

- **Significance in Programming:**
  - **Widely Adopted:** Popular in enterprise environments and large-scale applications.
  - **Community and Support:** Strong developer community and extensive documentation.
  - **Integration with .NET:** Seamless integration with the .NET Framework and ecosystem.

- **Relevance to Engineering:**
  - **Simulation and Modeling:** Building tools for simulating engineering processes.
  - **Data Analysis:** Developing applications for processing and analyzing engineering data.
  - **Automation:** Creating software for automating engineering tasks and workflows.

Part 2:

Understand C# Syntax, Data Types, and Control Structures

C# Syntax and Structure

ELE 503: Advanced Computer Programming and Statistics

- **Namespaces:** Organize code and prevent naming conflicts.
- **Classes:** Define objects and their behaviors.
- **Methods:** Encapsulate functionality within classes.
- **Main Method:** Entry point of the application.

```
using System;

namespace HelloWorldApp
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Hello, World!");
        }
    }
}
```

Explanation:

- **using System;**: Imports the System namespace, which contains fundamental classes.
- **namespace HelloWorldApp**: Defines a namespace for the application.
- **class Program**: Declares a class named Program.
- **static void Main(string[] args)**: Main method serving as the entry point.
- **Console.WriteLine("Hello, World!");**: Outputs text to the console.

Data Types in C#

ELE 503: Advanced Computer Programming and Statistics

- **Primitive Data Types:**

- **Integer Types:**
  - int: 32-bit signed integer.
  - long: 64-bit signed integer.

- **Floating-Point Types:**
  - float: 32-bit single-precision.
  - double: 64-bit double-precision.

- **Character Types:**
  - char: Single 16-bit Unicode character.

- **Boolean Type:**
  - bool: Represents true or false.

- **Non-Primitive Data Types:**

- **Strings:**
  - string: Represents a sequence of characters.

- **Arrays:**
  - Collections of elements of the same type.

Sample Code

```
int count = 10;
double temperature = 36.6;
char grade = 'A';
bool isActive = true;
string message = "Engineering Programming";
```

Variables and Constants

ELE 503: Advanced Computer Programming and Statistics

Variables:

•**Declaration:** Specify the data type followed by the variable name.

```
int number;
double price;
```

•**Initialization:** Assign a value to the variable

```
number = 5;
price = 99.99;
```

Constants:

•**Definition:** Immutable values that cannot be changed after declaration.

•**Declaration:** Use the const keyword

```
const double PI = 3.14159;
const string COMPANY_NAME =
"TechCorp";
```

Best Practices:

•Use descriptive names for variables and constants.

•Follow naming conventions (e.g., PascalCase for constants).

Operators in C#

ELE 503: Advanced Computer Programming and Statistics

- **Arithmetic Operators:**
  - + : Addition
  - - : Subtraction
  - * : Multiplication
  - / : Division
  - % : Modulus

- **Relational Operators:**
  - == : Equal to
  - != : Not equal to
  - > : Greater than
  - < : Less than
  - >= : Greater than or equal to
  - <= : Less than or equal to

- **Logical Operators:**
  - && : Logical AND
  - || : Logical OR
  - ! : Logical NOT

- **Assignment Operators:**
  - = : Assign
  - += : Add and assign
  - -= : Subtract and assign
  - *= : Multiply and assign
  - /= : Divide and assign

```
int a = 10;
int b = 5;
int sum = a + b;          // sum = 15
bool isEqual = (a == b); // isEqual = false
bool result = (a > b) && (b > 0); // result = true
```

Control Structures: If-Else Statements

ELE 503: Advanced Computer Programming and Statistics

- **Purpose:**
  - Execute code blocks based on certain conditions.
- **Syntax:**

```
if (condition)
{
    // Code to execute if condition is true
}
else if (anotherCondition)
{
    // Code to execute if anotherCondition is true
}
else
{
    // Code to execute if none of the above conditions are true
}
```

- Example: Temperature Check

```
double temperature = 37.5;

if (temperature > 38.0)
{
    Console.WriteLine("High fever detected.");
}
else if (temperature >= 36.5 && temperature <= 37.5)
{
    Console.WriteLine("Normal temperature.");
}
else
{
    Console.WriteLine("Low temperature.");
}
```

Control Structures: Loops

ELE 503: Advanced Computer Programming and Statistics

For Loop

•Purpose:

- Repeat a block of code a specific number of times.

•Syntax:

```
for (initialization; condition; iteration)
{
    // Code to execute
}
```

•Example: Temperature Check

```
for (int i = 1; i <= 5; i++)
{
    Console.WriteLine("Count: " + i);
}
```

While Loop

Purpose:

- Repeat a block of code as long as a condition is true.

•Syntax:

```
while (condition)
{
    // Code to execute
}
```

•Example: Decrementing a Counter

```
int count = 5;

while (count > 0)
{
    Console.WriteLine("Countdown: " + count);
    count--;
}
```

Control Structures: Loops

ELE 503: Advanced Computer Programming and Statistics

Do-While Loop

Purpose:

- Execute a block of code at least once and then repeat as long as a condition is true.
- **Syntax:**

```
do
{
    // Code to execute
}
while (condition);
```

- Example: User Input Validation

```
string input;
do
{
    Console.WriteLine("Enter 'yes' to continue:");
    input = Console.ReadLine();
}
while (input.ToLower() != "yes");
```

Control Structures: Switch Statements

ELE 503: Advanced Computer Programming and Statistics

Purpose:

- Select among multiple cases based on the value of a variable.
- **Syntax:**

```
switch (variable)
{
    case value1:
        // Code to execute for value1
        break;
    case value2:
        // Code to execute for value2
        break;
    default:
        // Code to execute if no case matches
        break;
}
```

- Example: Day of the Week

```
int day = 3;

switch (day)
{
    case 1:
        Console.WriteLine("Monday");
        break;
    case 2:
        Console.WriteLine("Tuesday");
        break;
    case 3:
        Console.WriteLine("Wednesday");
        break;
    case 4:
        Console.WriteLine("Thursday");
        break;
    case 5:
        Console.WriteLine("Friday");
        break;
    case 6:
        Console.WriteLine("Saturday");
        break;
    case 7:
        Console.WriteLine("Sunday");
        break;
    default:
        Console.WriteLine("Invalid day");
        break;
}
```

Methods and Functions in C#

ELE 503: Advanced Computer Programming and Statistics

Do-While Loop

Purpose:

- Encapsulate reusable code blocks to perform specific tasks.

Syntax:

```
returnType MethodName(parameters)
{
    // Code to execute
    return value; // if returnType is not void
}
```

- Example: Calculating the Area of a Circle

```
double CalculateArea(double radius)
{
    const double PI = 3.14159;
    double area = PI * radius * radius;
    return area;
}

// Usage
double r = 5.0;
double circleArea = CalculateArea(r);
Console.WriteLine("Area of the circle: " + circleArea);
```

Types of Methods:

- **Instance Methods:** Operate on instances of a class.
- **Static Methods:** Belong to the class itself rather than any particular instance.

Arrays and Collections

ELE 503: Advanced Computer Programming and Statistics

Arrays:

- **Definition:** Fixed-size, ordered collections of elements of the same type.
- **Declaration and Initialization:**

```
int[] numbers = new int[5];
numbers[0] = 10;
numbers[1] = 20;
// ...

// Or initialize with values
int[] scores = { 85, 90, 78, 92, 88 };
```

Multidimensional Arrays:

- Example: 2D Array for Matrix Representation

```
double[,] matrix = new double[3, 3]
{
    {1.0, 2.0, 3.0},
    {4.0, 5.0, 6.0},
    {7.0, 8.0, 9.0}
};
```

Collections:

- **List<T>:** Dynamic-size list

```
List<string> engineers = new List<string>();
engineers.Add("Alice");
engineers.Add("Bob");
```

Dictionary<TKey, TValue>: Key-value pairs.

```
Dictionary<string, double> measurements = new Dictionary<string, double>();
measurements.Add("Length", 12.5);
measurements.Add("Width", 7.8);
```

Object-Oriented Programming (OOP) in C#

ELE 503: Advanced Computer Programming and Statistics

- **Core Principles:**
  - **Encapsulation:** Bundling data and methods within classes.
  - **Inheritance:** Deriving new classes from existing ones.
  - **Polymorphism:** Ability to process objects differently based on their data type or class.
  - **Abstraction:** Hiding complex implementation details and exposing only necessary components.
- **Benefits in Engineering:**
  - **Modularity:** Easier maintenance and scalability of engineering software.
  - **Reusability:** Reuse existing classes and methods across multiple projects.
  - **Maintainability:** Simplifies debugging and updating codebases.

Classes and Objects

ELE 503: Advanced Computer Programming and Statistics

- **Classes:**
- **Definition:** Blueprint for creating objects.
- **Components:**
  - **Fields:** Variables to hold data.
  - **Properties:** Accessors for fields.
  - **Methods:** Functions to perform actions.
- **Objects:**
- **Definition:** Instances of classes.

- **Example: Defining and Using a Class**

```
class Engineer
{
    // Fields
    public string Name;
    public int ID;

    // Method
    public void DisplayInfo()
    {
        Console.WriteLine("Engineer Name: " + Name);
        Console.WriteLine("Engineer ID: " + ID);
    }
}

// Usage
Engineer eng = new Engineer();
eng.Name = "Charlie";
eng.ID = 101;
eng.DisplayInfo();
```

Inheritance and Polymorphism

ELE 503: Advanced Computer Programming and Statistics

Inheritance:

•**Purpose:** Allows a class to inherit properties and methods from another class.

•**Syntax:**

```
class SeniorEngineer : Engineer
{
    public string Department;

    public void DisplayDepartment()
    {
        Console.WriteLine("Department: " + Department);
    }
}

// Usage
SeniorEngineer senior = new SeniorEngineer();
senior.Name = "Diana";
senior.ID = 102;
senior.Department = "Aerospace";
senior.DisplayInfo();
senior.DisplayDepartment();
```

Polymorphism:

•**Definition:** Methods can have different behaviors based on the object that invokes them.

•**Example: Method Overriding**

```
class Engineer
{
    public virtual void Work()
    {
        Console.WriteLine("Engineer is working on projects.");
    }
}

class SeniorEngineer : Engineer
{
    public override void Work()
    {
        Console.WriteLine("Senior Engineer is leading the project.");
    }
}

// Usage
Engineer eng = new Engineer();
Engineer seniorEng = new SeniorEngineer();

eng.Work();          // Outputs: Engineer is working on projects.
seniorEng.Work();    // Outputs: Senior Engineer is leading the project.
```

Exception Handling

ELE 503: Advanced Computer Programming and Statistics

•**Purpose:** Manage runtime errors gracefully without crashing the program.

Syntax:

```
try
{
    // Code that may throw an exception
}
catch (ExceptionType ex)
{
    // Handle exception
}
finally
{
    // Code that runs regardless of exception occurrence
}
```

Example: Division by Zero

```
try
{
    int numerator = 10;
    int denominator = 0;
    int result = numerator / denominator;
    Console.WriteLine("Result: " + result);
}
catch (DivideByZeroException ex)
{
    Console.WriteLine("Error: Cannot divide by zero.");
}
finally
{
    Console.WriteLine("Operation completed.");
}
```

Introduction to the .NET Framework

ELE 503: Advanced Computer Programming and Statistics

- **What is .NET?**
  - A comprehensive development platform by Microsoft.
  - Provides a controlled environment for developing and running applications.

- **Components:**
- **Common Language Runtime (CLR):**
  - Executes C# programs.
  - Manages memory, security, and exception handling.

- **Base Class Library (BCL):**
  - Provides essential classes for tasks like file I/O, data manipulation, and more.

- **Languages:**
  - Supports multiple languages (C#, VB.NET, F#, etc.) interoperably.

- **Advantages:**
- **Cross-Platform Development:** With .NET Core and .NET 5/6+, develop applications for Windows, Linux, and macOS.

- **Robust Libraries:** Extensive built-in functionalities for various application needs.

- **Scalability and Performance:** Optimized for high-performance applications.

C# and .NET in Engineering Applications

ELE 503: Advanced Computer Programming and Statistics

- **Simulation Software:**
  - Developing tools for simulating physical systems and processes.
  - Example: Hydraulic system simulations, thermal analysis.
- **Data Analysis and Visualization:**
  - Processing large datasets and visualizing engineering data.
  - Example: Stress-strain analysis, performance metrics tracking.
- **Automation Systems:**
  - Creating software to automate repetitive engineering tasks.
  - Example: Automated testing systems, manufacturing process controls.
- **CAD and CAM Integration:**
  - Enhancing Computer-Aided Design (CAD) and Manufacturing (CAM) software with custom plugins and extensions.
- **Embedded Systems:**
  - Developing applications for embedded engineering devices and IoT solutions.

Part 3:

Write Simple C# Programs to Solve Engineering Problems

Sample C# Program: Hello World

ELE 503: Advanced Computer Programming and Statistics

Explanation:

1.using System;

- Imports the System namespace, which contains fundamental classes like Console.

2.namespace HelloWorldApp

- Defines a namespace to organize code and prevent naming conflicts.

3.class Program

- Declares a class named Program.

4.static void Main(string[] args)

- Main method serving as the entry point of the application.

5.Console.WriteLine("Hello, Engineering World!");

- Outputs the string to the console.

Sample C# Program: Engineering Calculation

ELE 503: Advanced Computer Programming and Statistics

Line-by-Line Explanation:

1.using System;

- Imports the System namespace for console operations.

2.namespace EngineeringCalculations

- Defines a namespace for engineering-related calculations.

3.class Program

- Declares the Program class.

4.static void Main(string[] args)

- Main method acting as the entry point.

5.Console.WriteLine("Triangle Area Calculator");

- Displays the program title.

6.Console.Write("Enter the base of the triangle (in meters): ");

- Prompts the user to input the base length.

7.double baseLength = Convert.ToDouble(Console.ReadLine());

- Reads and converts user input to a double.

8.Console.Write("Enter the height of the triangle (in meters): ");

- Prompts the user to input the height.

9.double height = Convert.ToDouble(Console.ReadLine());

- Reads and converts user input to a double.

10.double area = CalculateArea(baseLength, height);

- Calls the CalculateArea method to compute the area.

11.Console.WriteLine("The area of the triangle is: " + area + " square meters.");

- Outputs the calculated area.

12.static double CalculateArea(double baseLength, double height)

- Defines a method to calculate the area of a triangle.

13.return 0.5 * baseLength * height;

- Returns the computed area.

Hands-On Exercise 1 - Simple C# Program

ELE 503: Advanced Computer Programming and Statistics

Task: Write a C# Program to Calculate the Volume of a Cylinder

Instructions:

1.Define the Program Structure:

- Begin with using System;
- Define a namespace (e.g., CylinderVolumeCalculator).
- Create a Program class with a Main method.

2.Input Parameters:

- Prompt the user to enter the radius and height of the cylinder.
- Read and store the inputs.

3.Calculation:

- Use a method CalculateVolume to compute the volume using the formula V=πr2hV = \pi r^2 hV=πr2h.

4.Output the Result:

- Display the calculated volume.

Hands-On Exercise 2 - Control Structures

ELE 503: Advanced Computer Programming and Statistics

Task: Write a C# Program to Determine if a Number is Prime

Instructions:

1.Define the Program Structure:

- Begin with using System;
- Define a namespace (e.g., PrimeChecker).
- Create a Program class with a Main method.

2.Input:

- Prompt the user to enter an integer.

3.Prime Check Logic:

- Use a for loop to check divisibility from 2 to the square root of the number.
- Utilize if statements to determine if the number is prime.

4.Output the Result:

- Inform the user whether the number is prime or not.

Hands-On Exercise 3 - Object-Oriented Programming Basics

ELE 503: Advanced Computer Programming and Statistics

Task: Create a C# Class to Model an Engineering Student
Instructions:

1.Define the Class Structure:

- Create a Student class with properties: Name, ID, and GPA.
- Include a method DisplayInfo to print student details.

2.Instantiate and Use the Class:

- In the Main method, create an instance of Student.
- Assign values to the properties.
- Call the DisplayInfo method.

Part 4:

Explore the .NET Framework and Its Relevance to Engineering Applications

Exploring the .NET Framework

ELE 503: Advanced Computer Programming and Statistics

- **Components of .NET:**
  - **Common Language Runtime (CLR):** Executes C# programs, manages memory, and handles security.
  - **Base Class Library (BCL):** Provides essential classes for input/output, data manipulation, and more.
  - **Language Interoperability:** Allows different languages to work together seamlessly.

- **Benefits for Engineering Applications:**
  - **Rich Libraries:** Simplifies the development of complex engineering software by leveraging pre-built functionalities.
  - **Cross-Platform Support:** Develop applications that run on multiple operating systems.
  - **Scalability:** Build applications that can scale from small tools to large enterprise systems.

- **Example Libraries:**
  - **System.Math:** Mathematical functions.
  - **System.IO:** File input/output operations.
  - **System.Collections.Generic:** Data structures like lists and dictionaries.

Writing and Executing C# Programs in Visual Studio

ELE 503: Advanced Computer Programming and Statistics

Steps to Write and Execute a C# Program:

1.Install Visual Studio:

- Download and install Visual Studio Community Edition.

2.Create a New Project:

- Open Visual Studio.
- Select **"Create a new project"**.
- Choose **"Console App (.NET Core)"** or **"Console App (.NET Framework)"**.
- Name the project (e.g., CSharpIntro).

3.Write the Code:

- Replace the default code with your C# program.
- Utilize IntelliSense for code suggestions and error checking.

4.Build and Run:

- Press **F5** or click the **"Start"** button to compile and run the program.
- View the output in the console window.

5.Debugging:

- Set breakpoints by clicking in the margin next to the code.
- Use debugging tools to step through the code and inspect variables.

Visual Demonstration:

- **Screenshots or Live Demo:**
  - Show the Visual Studio interface.
  - Navigate through creating a project, writing code, and running the application.

Code Templates for Exercises

ELE 503: Advanced Computer Programming and Statistics

- Template 1: Volume Calculator
- Template 2: Temperature Converter

Instructions for Students:

- Complete the CalculateVolume and ConvertToFahrenheit methods.
- Test the programs with sample inputs to ensure accuracy.

Real-World Applications of C# in Engineering

ELE 503: Advanced Computer Programming and Statistics

1. Simulation Software

•**Example:** Developing simulation tools for fluid dynamics, structural analysis, and thermal systems.

•**Benefits:**

- Accurate modeling of complex engineering systems.
- Enables testing and optimization before physical implementation.

2. Data Analysis and Visualization

•**Example:** Creating applications to process and visualize large sets of engineering data.

•**Benefits:**

- Facilitates data-driven decision-making.
- Enhances understanding through graphical representations.

3. Automation and Control Systems

•**Example:** Building software for automating manufacturing processes or controlling machinery.

•**Benefits:**

- Increases efficiency and precision.
- Reduces human error and operational costs.

4. CAD/CAM Integration

•**Example:** Developing plugins or extensions for Computer-Aided Design (CAD) and Manufacturing (CAM) software.

•**Benefits:**

- Enhances functionality and customizability of design tools.
- Streamlines the design-to-production workflow.

5. Embedded Systems and IoT

•**Example:** Creating applications for embedded engineering devices and Internet of Things (IoT) solutions.

•**Benefits:**

- Enables connectivity and smart functionalities in engineering hardware.
- Supports real-time data collection and analysis.

Practical Considerations

ELE 503: Advanced Computer Programming and Statistics

- **Transition to Modern Languages:**
  - Understanding C# lays the foundation for learning other modern languages like Java, Python, and C++.
  - Emphasizes object-oriented and structured programming paradigms prevalent in current software development.
- **Interoperability:**
  - C# seamlessly integrates with other .NET languages and technologies, facilitating multi-language projects.
  - Supports calling native APIs and interfacing with hardware, crucial for engineering applications.
- **Performance and Scalability:**
  - Optimized for performance-critical applications in engineering.
  - Supports asynchronous programming and parallel processing to handle large-scale computations.
- **Community and Resources:**
  - Extensive online resources, libraries, and community support.
  - Continuous updates and improvements from Microsoft ensure C# remains relevant.

Homework Assignment

ELE 503: Advanced Computer Programming and Statistics

- **Programming Task:**
- **Objective:** Write and execute C# programs to solve engineering problems.
- **Tasks:**
  - **Cylinder Volume Calculator:**
    - Implement a program to calculate the volume of a cylinder.
    - Ensure the program handles user input validation.
    - Test the program with various inputs and report the results.
  - **Temperature Converter:**
    - Create a program to convert temperatures between Celsius, Fahrenheit, and Kelvin.
    - Include options for the user to select the conversion type.
    - Display accurate results based on user inputs.
- **Data Analysis Project:**
- **Objective:** Utilize C# to perform data analysis relevant to engineering.
- **Tasks:**
  - **Stress-Strain Analysis:**
    - Develop a program to calculate stress and strain based on user inputs for force and area.
    - Include calculations for Young's modulus.
  - **Energy Consumption Calculator:**
    - Design a program to compute electrical energy consumption based on user inputs for power, time, and efficiency.
    - Extend the program to generate a report summarizing the calculations.

Homework Assignment pt2

ELE 503: Advanced Computer Programming and Statistics

- **Historical Reflection:**
- **Objective:** Understand the evolution and impact of C# in engineering and software development.
- **Tasks:**
  - Write a short essay (300-500 words) on how C# has influenced modern programming languages.
  - Discuss specific features from C# that are present in today's languages.
  - Reflect on the importance of C# in developing engineering applications and maintaining legacy systems.