

# Module Introduction

Course Code: ELE1147

Course Name: Programming for Engineers

Credits: 15

Module Leader: Seb Blair BEng(H) PGCAP MIET MIHEEM FHEA

## Module Aims

This module aims to equip [you] with the skills to leverage programming languages effectively to address complex engineering challenges. It emphasizes utilizing both open-source and proprietary tools, mastering version control systems, and honing the ability to develop and manage codebases. [You] will enhance [your] proficiency in improving legacy code and integrating new features into existing systems.

# Module Learning Outcomes

On successful completion of this module a student will be able to:

- [1] Utilize programming languages proficiently to solve various non-trivial problems
- [2] Employ version control systems effectively to manage codebases, demonstrating competency in tasks such as branching, merging, and resolving conflicts to maintain code integrity
- [3] Evaluate and enhance existing codebases, identifying opportunities for improvement and implementing strategies to refactor legacy code while ensuring compatibility with new features to solve a complex engineering problem.
- [4] Demonstrate proficiency in adhering to industry best practices and considering scalability, maintainability, and performance

# Indicative Content

- Introduction to programming languages commonly used in engineering, such as Python, C, or C++, highlighting their strengths and applications in different domains.
- Overview of open-source and proprietary tools for software development, including Integrated Development Environments (IDEs), text editors, compilers, and debugging utilities
- Comprehensive exploration of version control systems such as Git, covering concepts like repositories, commits, branches, merges, and conflict resolution
- Practical exercises and case studies demonstrating the process of developing and managing codebases using version control, including collaborative workflows and code review practices
- Guidelines and best practices for adding new features to existing codebases, considering factors such as modularity, extensibility, and compatibility with legacy systems.
- Opportunities for practical implementation and experimentation, including coding exercises, projects, and simulations that allow students to apply theoretical concepts in practical scenarios

# Assessments ELEE1147

## 1. Logbook - 30%

- LO - 1,2.
- Pass mark - 40%
- [You] will be tasked with solving non-trivial problems using a variety of programming languages.

## 2. Coursework - 70%

- LO - 2,3,4.
- Pass mark - 40%
- Challenge based design and programming exercises that will need to be reported upon.