MEC2203: Computer Programming for Mechanical Engineers

Hours per semester				Weighted total mark	Weighted exam mark	Weighted continuous assessment mark	Credit unit
LH	PH	TH	CH				CU
45	30	75	60	100	60	40	4

Course description

Competency in a programming language is prerequisite to the study of computer engineering. Object-oriented programming, event-driven applications, and the use of extensive APIs (application programming interfaces) are fundamental tools that computer engineering students need early in their academic program.

Objectives

The objectives of this course are:

- To introduce the principles and fundamentals of computer programming.
- To equip the student with the skills of using a programming language to solve day to day problems.

Learning Outcomes

On completing this course the student should be able to:

- Describe how computer engineering uses or benefits from programming fundamentals.
- Identify the appropriate paradigm for a given programming problem.
- Use a suitable programming language to implement, test, and debug algorithms for solving simple problems.
- Describe the way a computer allocates and represents these data structures in memory.
- Outline the philosophy of object-oriented design and the concepts of encapsulation, sub classing, inheritance, and polymorphism.

Course Content

History and Overview **(4 Hours)**

- Indicate some reasons for studying programming fundamentals
- Influential people; important areas such as programming constructs, algorithms,

problem solving, data structures, programming paradigms, recursion, object- oriented programming, event-driven programming, and concurrent programming

- Contrast between an algorithm and a data structure
- Distinguish between a variable, type, expression, and assignment
- Highlight the role of algorithms in solving problems
- Describe some of the fundamental data structures such as array, record, stack, and queue
- Explain how divide-and-conquer strategies lend themselves to recursion
- Explore some additional resources associated with programming fundamentals
- Explain the purpose and role of programming fundamentals in computer engineering

Programming Languages **(4 Hours)**

- Definition and History
- Characteristics (Pragmatics, Semantics and Syntax)
- Distinction between Text-based and Visual Programming
- Classification (Categorical, Chronological and Generational)
- Comparison of common programming languages (C, C++, C#, Java)
- Programming errors and warnings (syntax, logical, etc.)

Programming Paradigms **(8 Hours)**

- Definition and rationale of a programming paradigm
- Types: Structured, Unstructured, Procedural, Object-oriented, Event-Drive, Generic etc.
- Separation of behavior and implementation

ISO/ANSI C++ Programming Fundamentals **(11 Hours)**

- Bjarne Stroustrup Design rules
- Console applications basics (Source file, Basic I/O, Standard I/O Consoles, Function main( )
- Fundamental data types
- Expressions and operators

- Control constructs (Conditional and Iterative)
- Pointers and Named collections (Arrays, Enumerators, Bit-fields, Unions)
- User-defined data types (Structures and Classes)
- Functions (In-built and User-defined)
- Object -oriented programming (Abstraction, Encapsulation, Inheritance, Composition, Polymorphism, Friend and Virtual Functions)
- File I/O

Algorithms and Problem-Solving **(8 Hours)**

- Problem-solving strategies
- The role of algorithms in the problem-solving process
- Implementation strategies for algorithms
- Debugging strategies
- The concept and properties of algorithms
- Structured decomposition

The Integrated Development Environment (IDE) **(6 Hours)**

- Definition
- Toolchains
- Advantages of IDEs
- Comparison of IDEs
- Using a typical IDE (Visual Studio)

Hands-on student activity (40 Hours)

Delivery Methods:

• The course will be taught by using lectures, tutorials, practical sessions involving hands-on project work and laboratories.

Assessment Methods:

• Course work (assignments and tests) and final examination and their relative contributions to final grade are shown as follows:

Requirement	**Percentage contribution**
Course work	40%

Final examination 60%

Total 100%

Recommended and Reference Books

Paul J. Lucas, The C ++ Programmer's Handbook**, Prentice Hall 1994. Jean Ettinger, Programming in C++ Macmillan Press,2003**

Deitel and Deitel **C++ How to program,** 4th Edition, Prentice Hall 2003.