Learning Outcomes:

	Learning Outcomes
1.	Design small computer programs as solutions to problems of low complexity.
2.	Implement the designs by writing well-structured programs that follow enforced programming language conventions and programming standards.
3.	Test, debug and document small computer programs.

Topics/Content/Outline

Topics include: classes, objects, methods, properties, data types decisions, iterations, arrays, text files, logic depiction.

Expanded Outcomes

Outcome 1:

Design small computer programs to problems of low complexity

- Discuss the steps in the program development cycle.
- Decompose a programming problem using a standard technique to represent object properties and behaviour.
- Select and use standard logic control structures to represent the flow of control of logic within methods (Range: sequence, selection, repetition)
- Read and express program logic using a relevant depiction method.
- Use simplified graphical notations for modelling components of a program.
- Desk-check the program logic of a given design
- Recognise and select design patterns and algorithms for simple problems from a given selection (Range: array manipulation, counting, addition, searching, data validation)

Outcome 2:

Implement the designs by writing well-structured computer programs that follow enforced programming language conventions and programming standards

- Translate designs to code using a standard programming language
- Identify and become familiar with the terminology used in the context of programming in this course
- Identify the data requirements of the program and translate them appropriately according to the conventions of the programming language
- Select fundamental data types fro variables and constants (Range: Primitive types i.e. integer, floating point, character and Boolean; User-defined types)
- Identify the standards and conventions that must be followed when writing programs
- Apply the appropriate basic operations in the implementation of the program design (Range: assignment, input and output, evaluation of numerical and logical expressions).
- Apply the appropriate logic control structures in the implementation of the program design (Range: sequence, selection, iteration)
- Use appropriate data organisation and data types to code data (Range: Primitive types; Reference types; Fixed size collections like an array of simple data-types).
- Use appropriate class libraries.
- Use text files for input and output.

Outcome 3:

Test, debug and document small computer programs

- Compile, test and debug programs using a test plan until free of errors.
- Use an appropriate debugging tool.
- Validate the program as a solution to the original problem.
- Document the program to a given standard