Subject Description Form

Subject Code	ENG2002
Subject Title	Computer Programming
Credit Value	3
Level	2
Pre-requisite/Co-requisite/Exclusion	Nil
Objectives	(i) To introduce the fundamental concepts of computer programming. (ii) To equip students with solid skills in Python programming. (iii) To equip students with techniques for developing structured and object-oriented computer programs. (iv) To demonstrate the techniques for implementing engineering applications using computer programs.
Intended Subject Learning Outcomes	**Upon completion of the subject, students will be able to:** 1. Familiarize themselves with at least one Python programming environment. 2. Be proficient in using the basic constructs of Python to develop a computer program. 3. Develop a structured and documented computer program. 4. Understand the fundamentals of object-oriented programming and be able to apply it in computer program development. 5. Apply computer programming techniques to solve practical engineering problems.
Subject Synopsis/ Indicative Syllabus	**Syllabus:** 1. **Introduction to Programming** Components of a computer; Data representation in computers; Programming environment; Python IDE; Editing, saving, and running a script; Process of application development. 2. **Bolts and Nuts of Python** Data types; Variables and constants; Operators, expressions, and statements; Basic syntax; Functions and modules; Scope of variables; Python modules; Absolute and relative import. 3. **Program Flow Control and Functions** Branching and looping; Iterators; Unicode; Python functions; static functions; Lambda function; Position arguments and default arguments; args and kwargs; Interface with command line; argparse 4. **Program Design and Debugging** Structured program design; Testing and debugging a program; Exception and assertion. 5. **Strings and File I/O** String encoding format; F-string; String operations; String and number conversion; File and directory manipulations; The “os”, “sys”, and “shutil” modules; Reading/writing text and numbers from/to a file. 6. **Tuples, Lists, Dictionaries, and Sets** Basic tuple and list operations; Searching and sorting lists; Dictionary literals; Basic dictionary operations; Built-in tuple/list/dictionary/set methods and functions; Use of enumerate and zip 7. **Basic Object-Oriented Programming**

Objects and classes; Attributes and methods; Inheritance and polymorphism; Special methods and operator overloading.

8. **Data Analytics with Python Libraries**
   Introduction to NumPy, Pandas, and Matplotlib; NumPy arrays, built-in methods, and mathematical operations; Reading/writing data files using Pandas; Pandas operations and functions; Data visualization with Matplotlib

Teaching/Learning Methodology

Teaching and Learning Method	Intended Subject Learning Outcome	Remarks
Lectures, supplemented with short quizzes	2,3,4	Students are introduced to the knowledge of computer programming through explanation and illustrative examples. Comprehension of the knowledge is strengthened with short quizzes. Students will be able to monitor the skills of using Python and apply the techniques of developing structured object-oriented applications.
Laboratories/tutorials where problems are given to students for them to solve	1,2,3,4	Students apply what they have learnt in lectures and solve problems in exercises. The purpose is to ensure students have captured the important points. Tutors will aid the lecturer in helping the students finishing the exercises, and interactive Q&A will take place.
Assignment, tests and final examination	1,2,3,4,5	By doing assignment, students will develop a firm understanding and comprehension of the knowledge taught. They will analyse given Python applications and apply knowledge to solve problems. They will have to design solutions by evaluating different alternatives. To enhance the students' problem-solving skill in a given programming environment, open-book programming tests are arranged regularly. To assure students' understanding of fundamental concepts, a closed-book final examination is arranged.

Assessment Methods in Alignment with Intended Learning Outcomes

Specific Assessment Methods/Tasks	% Weighting	Intended subject learning outcomes to be assessed				
		1	2	3	4	5
1. In-class exercises and homework	10%	✓	✓	✓	✓	
2. Short-quizzes	10%		✓	✓	✓	
3. Programming tests	30%	✓	✓	✓	✓	✓

	4. Assignment	20%	✓	✓	✓	✓	✓
	5. Final examination	30%	✓	✓	✓	✓	✓
	Total	100%					

Explanation of the appropriateness of the assessment methods in assessing the intended learning outcomes:

The short-quizzes are for assessing the understanding of fundamental concepts. The in-class exercises and homework are conducted to help students familiarized with the programming language and skills. The programming tests are for assessing the ability of students on solving computer problems through programming within a specified period. Through doing assignments, students will be able to experience how to solve engineering problems and design solutions by using a systematic approach. The final examination is for assessing the students' ability on using the programming language and analysing computer programs.

Student Study Effort Expected	**Class contact:**	
	• Lectures, Tests and Quizzes	26 Hours
	• Laboratory/Tutorial	13 Hours
	Other student study effort:	
	• Self-studying	57 Hours
	• Homework	12 Hours
	Total student study effort:	**108 Hours**

Reading List and References

Reference Books:

1. G. van Rossum and the Python development team, *Python Tutorial Release 3.10.0*, Nov. 2021.
2. C. Hill*, Learning Scientific Programming with Python*, (2nd ed.) Cambridge: Cambridge University Press, 2020.
3. C.P. Millike, *Python Projects for Beginners: a ten-week bootcamp approach to Python programming*. Berkeley, CA: Apress, 2020.

January 2023