CENG109-Programming and Computation 1

Course Code	Course Name	Semester
CENG109	Programming and Computation 1	Fall ☒ Spring ☒ Summer □

Hours			Credit	ECTS
Theory	Practice	Lab	3	6
3	0	0		

Course Details	
Department	Computer Engineering
Course Language	English
Course Level	Undergraduate ☒ Graduate □
Mode of Delivery	Face to Face ☒ Online ☒ Hybrid ☒
Course Type	Compulsory ☒ Elective □
Lecturer(s)	
Course Objectives	This course introduces fundamental programming concepts and computational thinking. It is designed to be language-agnostic, allowing instructors to teach using their preferred programming language, such as C or Python. The course covers basic programming constructs, problem-solving techniques, and introductory algorithms and data structures.
Course Content	This course introduces students to the fundamentals of programming, starting with an overview of programming concepts and setting up the development environment. Students will explore block-based programming using Scratch, progressing from basic to advanced concepts, including event-driven programming. The course transitions into foundational problem-solving techniques through pseudocode and flowcharts. Core programming principles such as variables, data types, mathematical expressions, arrays, control flow constructs, loops, and functions are covered in-depth. Advanced topics include recursion and an introduction to object-oriented programming (OOP), emphasizing classes, inheritance, and polymorphism through practical implementation. The course integrates hands-on projects and tutorials to reinforce learning, culminating in comprehensive reviews to prepare for mid-term and final assessments.
Course Method/ Techniques	Lecture ☒ Question & Answer ☒ Presentation □ Discussion ☒
Prerequisites/ Corequisites	-
Work Placement(s)	-

Textbook/References/Materials

- C: How to Program, International Edition H. Deitel, P. Deitel, Prentice Hall
- Introduction to Programming in Python: An Interdisciplinary Approach / Robert Dondero, Kevin Wayne, Robert Sedgewick

Course Category				
Mathematics and Basic Sciences	☒		Education	☐
Engineering	☒		Science	☐
Engineering Design	☒		Health	☐
Social Sciences	☐		Profession	☒

Weekly Schedule

No	**Topics**	**Materials/Notes**
1	**Introduction to Programming** · Introduction to the course and syllabus overview. · What is a program? What is a programming language? · Historical context and evolution of programming languages. · Tutorial on setting up the programming environment and IDE installation.	Lecture notes, textbooks
2	**Scratch - Basics of Block-Based Programming** · Introduction to Scratch or Blockly. · Understanding basic programming concepts using block-based programming. · Creating simple projects to illustrate basic concepts.	Lecture notes, textbooks
3	**Scratch - Advanced Concepts** · Developing more complex projects using Scratch or Blockly. · Introduction to event-driven programming. · Transitioning from block-based to text-based programming.	Lecture notes, textbooks
4	**Introduction to Pseudocode and Flowchart** · Introduction to Pseudocode and Its Importance · Writing Algorithms in Pseudocode · Translating Pseudocode into a Programming Language ·	Lecture notes, textbooks
5	**Variables and Data Types** · Understanding variables, constants, and data types. · Declaring and using variables in a chosen programming language. · Tutorial on variable declarations and type usage.	Lecture notes, textbooks
6	**Mathematical Expressions** · Arithmetic operators and their usage. · Writing and evaluating mathematical expressions.	Lecture notes, textbooks

	· Understanding operator precedence. · Tutorial on constructing mathematical expressions.	
7	**Arrays** · Introduction to arrays and their significance. · Declaring, initializing, and accessing array elements. · Tutorial on basic array operations.	Lecture notes, textbooks
8	**Mid-Term**	
9	**Control Flow Constructs: sequence, selection, and repetition** **Conditional Statements** · In-depth look at if, else if, else statements. · Writing nested conditional statements. · Tutorial on using conditional statements to solve problems.	Lecture notes, textbooks
10	**Loops** · Introduction to loops: while, for, and do-while loops. · Using loops for iteration and repetitive tasks. · Tutorial on loop control statements: break and continue.	Lecture notes, textbooks
11	**Functions** · Understanding function definition and invocation. · Parameters and return values in functions. · Tutorial on writing and using functions effectively.	Lecture notes, textbooks
12	**Recursion** · Basic concepts of recursion. · Writing and understanding recursive functions. · Examples and applications of recursion. · Tutorial on debugging recursive functions.	Lecture notes, textbooks
13	**Object-Oriented Programming – Fundamentals – 1** Class Fundamentals • Introduction to classes and objects. • Understanding the concept of instances. • Creating and using classes in a chosen programming language. • Practical examples of class implementation. Inheritance • Understanding inheritance and its importance. • Implementing inheritance in practice. • Examples of single and multiple inheritance. • Tutorial on creating and using derived classes.	Lecture notes, textbooks
14	**Object-Oriented Programming – Fundamentals – 2** Polymorphism • Introduction to polymorphism and its benefits. • Implementing polymorphism with method overriding. • Examples of polymorphism in a chosen programming language. • Practical applications of polymorphism. Practical Implementation	Lecture notes, textbooks

	- Combining classes, inheritance, and polymorphism in a project. - Step-by-step guide to designing a simple OOP-based project. - Hands-on practice with real-world scenarios. - Debugging and testing OOP code.	
15	**Finals** · Comprehensive review of all course materials. · Addressing student questions and clarifying concepts. · Preparation for the final exam.	Lecture notes, textbooks
16	**Final Exam**	

Assessment Methods and Criteria		
In-term studies	**Quantity**	**Percentage**
Attendance		
Lab		
Practice		
Fieldwork		
Course-specific internship		
Quiz/Studio/Criticize	1	10
Homework	4	20
Presentation / Seminar		
Project		
Report		
Seminar		
Midterm Exam	1	20
Final Exam	1	50
	Total	**100%**
Contribution of Midterm Studies to Success Grade		
Contribution of End of Semester Studies to Success Grade		
	Total	**100%**

ECTS Allocated Based on Student Workload			
Activities	**Quantity**	**Duration (Hrs)**	**Total Workload**
Course Hours	14	3	42
Lab			
Practice			
Fieldwork			
Course-specific Work Placement			
Out-of-class study time	14	3	42
Quiz/Studio/Criticize			
Homework	4	3	12

Presentation / Seminar			
Project			
Report			
Midterm Exam and Preparation for Midterm	1	25	25
Final Exam and Preparation for Final Exam	1	30	30
Total Workload			**151**
Total Workload / 25			**6.04**
ECTS Credit			**6**

Course Learning Outcomes	
No	**Outcome**
L1	An ability to apply knowledge of science, mathematics, and engineering.
L2	An ability to design programs and algorithms
L3	An ability to work with multi-disciplinary teams.
L4	An ability to identify, formulate, and solve engineering problems.
L5	Take responsibility to solve unpredictable and complex problems encountered in applications as an individual and as a member of a team
L6	Plan and manage activities in teamwork
L7	An ability to use the techniques, skills, and modern engineering tools necessary for engineering practice.
L8	Can do research on interdisciplinary fields.

Contribution of Course Learning Outcomes to Program Competencies/Outcomes

Contribution Level: 1: Very Slight, 2: Slight, 3: Moderate, 4: Significant, 5: Very Significant

	P1	**P2**	**P3**	**P4**	**P5**	**P6**	**P7**	**P8**	**P9**	**P10**	**P11**					**Total**
L1	5	4	3	4	3	2	1	4	2	2	2					32
L2	4	5	5	4	3	2	1	3	2	2	2					33
L3	2	3	3	3	2	5	3	3	3	3	2					32
L4	4	5	4	4	3	3	2	4	3	3	3					38
L5	3	4	3	3	3	5	3	4	4	4	3					39
L6	2	3	3	3	2	5	3	4	3	3	3					34
L7	4	4	4	5	4	3	3	4	3	4	3					41
L8	4	4	3	4	5	3	2	5	4	3	4					41
															Total	290

i. Adequate knowledge in mathematics, science, and subjects specific to Computer Engineering; ability to use theoretical and applied knowledge in these areas to solve complex engineering problems.

ii. Ability to identify, formulate, and solve complex engineering problems; ability to select and apply appropriate analysis and modeling methods for this purpose.

iii. Ability to design a complex system, process, device, or product under realistic constraints and conditions to meet specific requirements; ability to apply modern design methods for this purpose.

iv. Ability to develop, select, and use modern techniques and tools required for the analysis and solution of complex problems encountered in engineering practice; ability to use information technologies effectively.

v. Ability to design and conduct experiments, collect data, analyze and interpret results in order to investigate complex engineering problems or research topics specific to the discipline of Computer Engineering.

vi. Ability to work effectively in disciplinary and multidisciplinary teams; ability to work individually.

vii. Ability to communicate effectively in oral and written Turkish; knowledge of at least one foreign language; ability to write effective reports and understand written reports, to prepare design and production reports, to make effective presentations, to give and receive clear and understandable instructions.

viii. Awareness of the necessity of lifelong learning; the ability to access information, to follow developments in science and technology and to continuously renew oneself.

ix. Acting in accordance with ethical principles, professional and ethical responsibility awareness; knowledge of standards used in engineering applications.

x. Knowledge about business life practices such as project management, risk management, and change management; awareness of entrepreneurship, innovation; knowledge about sustainable development.

xi. Knowledge about the effects of engineering applications on health, environment, and safety in universal and social aspects and the problems of the age reflected in the field of engineering; awareness of the legal implications of engineering solutions.