Undergraduate Course

Department of Industrial Engineering and Engineering Management

National Tsing Hua University

Course No.	10620IEEM214000	**Required/Elective course**		Elective	
Course Time	W7W8W9 FAFBFC	**Room**	R827	**Size limit**	50
Credits	3			**Hours offered per week**	3
Math	**Basic Science**	**Engineering**			
		Theory	**Design**		
1	0	1	1		
Course Title	□□□□□□□ Programming Design and Application				
Lecturer	Dr. Hareesh □□ (hareesh.pillai@ie.nthu.edu.tw)				
TA	Ms. Yeh Li-chia □□□ (home.yeh@gmail.com)				
Prerequisite	No prerequisite required. The course will have basic computation concepts refresher in the first week.				

Core capability to be cultivated by this course	■	Integration-□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□ Abilities of integrating various technologies of Industrial Engineering and Engineering Management	20%
	■	Information -□□□□□□□□□, □□□□□□□□□□□ Utilization of information technology to problem solving and applications in the field of Industrial Engineering	30%
	■	Interaction-□□□□□□□□□□□□ Coordination and communication abilities with teamwork	20%
	■	Innovation/Ideas-□□□□□□□□□□□□ Creative and innovative capabilities	10%
	■	Internationalization-□□□□□□□□□□ Global interaction and public presentation skills	20%
Course Description	This course is designed for **undergraduate** students. Software development is the process of conceiving, specifying, designing, programming, documenting, testing, and bug fixing involved in creating and maintaining software components. The Programming Design and Application course is focused on design for software programming using Unified Modeling Language outlined in ISO/IEC 19501:2005 standard and application of the generated design using programming paradigms such as procedural (C programming), object-oriented (Java), and web (PHP). Software design involves problem-solving including at high-level		

	problem abstraction and algorithm design. A programming language is a vocabulary to translate this design into implementation. A successful implementation requires understanding a host of design tools, constraints, programming methodologies, and data storage components. This course is designed to understand and practice a selected variety of such components with an emphasis on real-world applications to engineering problems.
Textbook	• Kanetkar, Y. P. (2016). *Let us C*. BPB publications. • Date, C. J. (1997). *A guide to the SQL standard: a user's guide to the standard database language SQL*. Addison-Wesley Professional.
References	• Pressman, R. S. (2005). *Software engineering: a practitioner's approach*. Palgrave Macmillan. • Martin, J., & Odell, J. J. (1994). *Object-oriented methods*. Prentice Hall PTR. • ISO/IEC 19501:2005 - Information technology open distributed processing Unified Modeling Language (UML) Version 1.4.2 • https://www.w3schools.in/java-tutorial/ • https://www.w3schools.com/php/
Teaching Method	The primary teaching objective will be towards enabling students to think logically and in the context of real-world design and applications. The course is planned as a practice centered applications course.
Teaching software	PowerPoint, Turbo C/ GCC, Java development kit, draw.io, WAMP
Syllabus	**Week 1. Class Introduction and Overview:** **Week 2. Programming Logic and Techniques:** I-P-O cycle, Understanding the difference between code, program, and software, the role of design in programming, software development life cycle, flow charts, integrated development environments, rapid application development. **Week 3. Procedural Programming using C:** Overview of programming paradigms, C programming, running C programs, the structure of C programs, C's standard libraries, data types, language constructs, arrays, pointers, structures, unions. **Week 4. Lab Practice Session:** **Week 5. Fundamentals of Object-Oriented Design:** Fundamental concepts of object orientation: object, class, abstraction, interface, implementation, aggregation, composition, generalization, sub-class and polymorphism, architecture style, Object Oriented Methodology (OOM), the advantage of OOM, OOP concepts with Java programming. **Week 6. Student Project Preliminary Discussion:** **Week 7. Lab Practice Session:** **Week 8. Programming Design with UML:** Visual modeling, component and model-driven development using Unified Markup Language (UML), structural UML diagrams (class diagram, package diagram, object diagram, component diagram, composite structure diagram, deployment diagram),

	behavioral UML diagrams (activity diagram, sequence diagram, use case diagram, state diagram, communication diagram, interaction overview diagram, timing diagram). **Week 9. Midterm:** **Week 10. Structured Query Language using MySQL:** Storage in I-P-O cycle, the role of data in software, modeling data using data flow diagram, ER modeling, relational database using open source MYSQL, data definition language (create, alter, drop), data manipulation language (Select INSERT, UPDATE and DELETE). **Week 11. Lab Practice Session:** **Week 12. Web Programming using WAMP:** Web distribution, HTML, HTML 5, static rendering, dynamic rendering, just in time pages (JIT), session management concepts, 3-tier architecture, fat client, fat server, setting up a web server using Windows Apache MySQL and Hypertext Preprocessor (WAMP), PHP script, syntax, tags. **Week 13. Lab Practice Session and Project Discussion:** **Week 14. Programming with Database Connectivity:** File handling in C programming, programming for database connectivity, java database connectivity (JDBC), windows open database connectivity (ODBC), database access concepts through PHP. **Week 15. Lab Practice Session/ Optional Final Team Presentation:** **Week 16. Applications Programming Interface (API) Programming:** API basics, the API economy, API infrastructure, networking layers, cloud computing disruption, API implementation, control, security, integration benefits, interoperability, database connectivity. **Week 17. Lab practice session and Team Project Update/ Optional Final Team Presentation:** **Week 18. Team Project Final Presentation:**
Evaluation	The course includes weekly programming practice homework and sessions mainly to apply concepts discussed in class, review material presented in class, and engage students in mini-quizzes in class. Along with these small quizzes, there is a midterm written exam, and a final project. Grades will be based on individual and project performance. Individual grades come from class attendance, homework performance, a midterm examination, and a final group project. **This is an English taught course, student presentation in English is highly recommended alternatively oral presentation is Chinese allowed, however the PowerPoint slides and final report word file has to be in English only.** Attendance: 10% Class interactions: 10% Midterm: 25% Homework: 30%

	Group project – 25%
Course website	Lecture notes (TBU)