INSTITUTE OF CIVIL ENGINEERING
College of Engineering
University of the Philippines, Diliman, Quezon City

CE 25 (MATHEMATICAL METHODS IN CIVIL ENGINEERING II)

COURSE GUIDE

WHAT IS THIS COURSE?

Welcome to CE 25 mathematical Methods in Civil Engineering II!

This is the second course in a series of courses that will equip students with skills that will help them solve problems encountered in major courses in the BS Civil Engineering program. In CE 24, you learned about various analytical methods to solve commonly encountered engineering problems. In this course, you will learn about **basic programming** and **numerical methods** for solving mathematical problems and problems governed by differential equations.

The topics you will learn from this course will help you solve civil engineering problems, which are too complex to solve analytically, using numerical techniques. This course will also enable you to write computer programs that can assist you in solving these problems. The general-purpose and procedural programming language, *C*, will be used in the lectures and exercises but the algorithms that you will learn can easily be used with any language of your choice in the future.

WHAT SHOULD YOU GAIN FROM THIS?

After completing this course, you should be able to—

1. (**CLO1**) Apply computational skills in solving mathematical problems in civil engineering.
2. (**CLO2**) Formulate a numerical model that will approximate the solution of ordinary and partial differential equations.
3. (**CLO3**) Develop programs that will facilitate solving mathematical problems in civil engineering numerically.

WHAT WILL YOU BE LEARNING?

A. C Programming Concepts	**B. Numerical Methods**
1. Introduction a. Computers & Operating Systems* b. C vs other programming languages* 2. Basic C Programming a. *main* Function and the Anatomy of a C-program b. Build Process and Debugging c. I/O Streams and Files d. Basic Data Types and Operations 3. Control Structures a. Selection Control using *if-else* and *switch* b. Repetition Control using *for*, *while* and *do-while* 4. Functions a. Standard Library and User-Defined Functions b. Declarations, Definitions, and Multiple Source/Header Files c. Scope and Lifetime of Variables	1. Introduction – Numerical Methods 2. Root-Finding Methods a. Fixed-point Iteration b. Newton-Raphson Method c. Bracketing Methods 3. Systems of Linear and Non-linear equations a. Direct Methods$^+$ b. Iterative Methods 4. Interpolation, Approximation, and Curve-Fitting a. Direct Method b. Lagrange and Newton Polynomials c. Least Squares Method d. Data Linearization 5. Numerical Differentiation and Integration a. Difference Formulas b. Higher-ordered Derivatives

5. Arrays a. 1-D Arrays and Strings b. Multi-dimensional Arrays c. Passing Arguments to Functions 6. Pointers a. Addresses, Values, and Pointer Operations b. Passing parameters by-value vs. by-reference c. Dynamic Memory Allocation 7. Structures*	c. Newton-Cotes Quadrature Formulas d. Multiple Integrals 6. Numerical Solutions of Ordinary Differential Eqns a. Initial Value Problems i. First-order Equations ii. Higher-order and System of Equations b. Boundary value problems 7. Numerical Solutions to Partial Differential Equations a. Finite Difference Method for Hyperbolic and Parabolic PDEs b. Finite Difference Method for Elliptic PDEs: Direct and Iterative Methods

$^{+}$Topics to be reviewed
* Items marked by (*) are optional topics that will be discussed when time permits

HOW WILL THIS COURSE BE DELIVERED?

Course Site

This course will be delivered remotely. Also, it will be **a**synchronous by default. All learning materials will be made available in our course page in the **University Virtual Learning Environment (UVLE)** website (or any other viable platform for all concerned).

You will be automatically enrolled by your instructors in our course page. If you have not done this yet, please change your email under Preferences > Edit Profile to your **@up.edu.ph** email.

Communication Plan

Communication with your instructors will be through the following means:

- Synchronous Sessions
  To facilitate consultations, demonstrations, feedback, and student-to-student discussions, a Zoom meeting room will be opened every lecture session of the week (**Tuesday, 1-3 PM**). The meeting IDs and passwords for these sessions are:
  Meeting ID: **554 504 5694**
  Password: **Num3r1cal**
  Attendance in Zoom sessions is not required but highly encouraged. The meeting will be open until 4PM if at least one student joins. If nobody attends the session, the meeting will be closed at 1:30 PM in order to save resources but will be opened again once students decide to drop by any time before 4PM. (Those who need to consult about the scheduled topics will just need to contact me before entering the meeting.)
- Course Page (Announcements and Fora)
  Announcements and updates relevant to the course will be posted in our UVLE course page through the News Forum as often as necessary. Everyone, at least those with consistent internet access, should subscribe to this forum so you can receive email notifications whenever there are new posts.
- Messenger and Email
  The preferred communication channel is Messenger because it is available even to those with limited internet access or on Free Data. It also provides a convenient way of tracking messages and replying to specific questions. Formal concerns and inquiries that need to be forwarded to other offices must be sent through email.

Teaching Strategies and Learning Activities

In the handling of this course, you are at the center of the learning process. Learning activities are designed based on this principle to make you engage with the material, with your classmates, and with me. As the student, you are primarily responsible for your own learning. Different activities, both formative and summative, are designed for you to demonstrate these learnings.

The numbered topics listed in the course outline above will be contained in a **module**. These modules are then grouped into **clusters**. Each cluster has an estimated completion time window which further breaks down into time windows for each module contained in it.

Please see the **How is Learning Scheduled?** section below for details.

Assessment Strategies and Activities

"Not everything that matters can be counted, and not everything that can be counted matters."
- attributed to Albert Einstein

All activities are designed to facilitate learning or self-reflection, but not everything will be. Formative assessments will not be included in your final grade. These include **Worksheets** which are included in each module and are designed for you to evaluate your learning of the subject matter. On the other hand, there will be summative assessments in the form of **Machine Exercises (MXs)** and **Problem Sets**. MXs will test your programming skills while Problem Sets will test your understanding and proficiency in carrying out detailed manual solution of the numerical methods. Please see the **How will you be Graded?** section for details.

WHAT COURSE MATERIALS WILL YOU BE USING?

Programming software

This course requires an Integrated Development Environment (IDE) - a software that lets you write and edit codes, as well as compile and run them. Any IDE is okay, but I strongly recommend for each operating system the following:

- *Windows*: **Visual Studio 2019** *Community Edition* from https://visualstudio.microsoft.com/**.** Choose *Desktop* in app type, *C++* in technology, and *Windows* on dev box os options.
- *Mac:* download **Xcode** from the App Store.
- *Linux*: **Visual Studio Code** or simply ***gcc***.
  (Other cross-platform options: Visual Studio Code, Code::Blocks, CLion)
- *iOS*: download **Mobile C** (C/C++ Compiler) from the App Store.
- *Android*: you choose between the following applications which are all available in Google Play Store:
  - **Cxxdroid** or **CppDroid** - free apps.
  - **CppDroid** (C/C++ IDE) - a better, but paid app that costs Php 209.

If you are planning to use a different software from those stated above, please tell me IMMEDIATELY after reading this document.

Course Pack

To facilitate your learning process, we have prepared a course pack which consists of the following:

- **Course guide**
  This is the document you are currently reading, which provides you important information on the major aspects of the course.
- **Study Guides**
  This is divided into multiple contents – designed so each will not take too much time to study so you can pace yourself without being overwhelmed with the amount of information. Recorded videos discussing the contents are also available in the course page. In the modules about C programming, source codes of the example programs shown in the lecture notes and videos will be available for download in our course page.
- **Worksheets**

A **Worksheet** is a hands-on exercise guide that contains: (1) example source codes which you can run and play with, (2) debugging exercises where you will identify errors in code/code snippets and try to fix them, and (3) bonus knowledge/topics that are not included in the study guides.

- **Machine Exercise guides**
  At the end of each cluster, you will be tasked to write a solve engineering mathematical problems that will demonstrate your proficiency in both programming and numerical method being studied. These individual computer exercises are called **Machine Exercises**. The machine exercise guides contain problem statement and the specifications – input/s, process/es, and output/s – of the computer program you are tasked to create. If applicable, example inputs and outputs are also included so you can test the functionality of your program.

All these materials are accessible in our course page in UVLE. To those students with very little to no reliable access to the internet, a package of printed versions of all course materials will be sent to their mailing address by courier upon their request.

I highly recommend that you read each guide first before you read/watch/listen to any learning resource, so that you have a clear idea on what lies ahead and what is expected of you.

Main References

1. Deitel, P., Deitel, H. C: How to Program, 8th ed. (2015). Pearson Prentice Hall, Upper Saddle River.
2. Kreyszig, E. Advanced Engineering Mathematics (2011). Wiley, Hoboken, N.J.
3. Hoffman, J. D. (2001). Numerical methods for engineers and scientists. New York: Marcel Dekker.
4. Chapra, S.C. and Canale, R.P., Numerical Methods for Engineers, 7th ed.
5. O'Neil, P.V., Advanced Engineering Mathematics, 8 th Ed.
6. Henner, V. Belorezova, T., Khenner M., Ordinary and Partial Differential Equations

HOW IS LEARNING SCHEDULED?

Please note that according to the UP Academic Credit Transfer System (ACTS), 1 academic credit (unit) = 38-48 hrs of student workload (including 13-16 hours of academic instruction). Thus, 3 academic credits (units) = (a minimum of) 114 hours of student workload for the semester. Because CE 25 is equivalent to 3 academic units and this semester lasts 14 weeks, **this translates to a minimum of approximately eight (8) hours of student workload every week**.

There will be weeks where the allotted topics will be relatively "lighter" or "heavier" that others but rest assured that no weekly workload will exceed eight (8) hours. This is in consideration of other requirements you will be accomplishing from other courses.

Although the tasks are distributed so you can finish them at a scheduled time, this course is still designed to be **self-paced**, which means that you can go through it as quickly or as slowly as you wish. This is in consideration of our extraordinary circumstances. Therefore, do not hesitate to consult on ANY TOPIC ANY TIME during the semester.

The following table lists the topics to be discussed per week and their respective target earning outcomes. The clusters and modules are indicated as well as the schedule when the Machine Exercises and Problem Sets will be given.

Week	Cluster	Module	TOPICS	Target Learning Outcomes	MX	PS
1		0	Course Intro	❑ Identify course expectations of teacher and students ❑ Introduce to numerical methods, computers, and programming		
		1	A1/B1. Introduction	❑ Discuss hardware and software relationships and data hierarchy. ❑ Identify the advantages of C over other programming languages		
2		2	A2. Basic C Programming	❑ Indentify parts of C-programs ❑ Discuss the different stages of program development. ❑ Develop a program that makes use of preprocessor directives		
			a. *main()* and the Anatomy of a C-program			
			b. Build Process and Debugging Concepts			
		3	c. I/O Streams and Files	❑ Discuss basics of file-handling ❑ Discuss how to read from or write to a file		
	1	4	d. Basic Data Types and Operations	❑ Define simple data types and use variables in basic arithmetic operations.		
3		5	A3. Control Structures	❑ Explain how logical expressions are evaluated in C ❑ Create flowcharts and pseudocodes to represent an algorithm. ❑ Develop a program that uses if-else and switch statements, to execute a decision point in an algorithm.		
			a. Selection Control Structures			
		6	b. Repetition Control Structures	❑ Distinguish the types of loop structure to use in a particular problem. ❑ Develop a program that uses the while, for, and do statements, to execute a part of an algorithm that requires iteration. ❑ Implement the break and continue statements to alter the flow of control in a program.	1	
4		7	A4. Functions	❑ Explain how functions work in C ❑ Develop a program that makes use some common library functions and user-defined functions. ❑ Discuss the different scopes of an identifier and lifetime of variables ❑ Develop a program that uses recursive functions to solve a problem		
			a. Standard Library			
		8	b. User-defined Functions			
	2		c. Scope and Lifetime of Variables			
5		9	B2. Root-Finding Methods	❑ Approximate the root/s of an equation using open-domain methods ❑ Approximate the root/s of an equation using closed-domain methods ❑ Develop a program that implements a root-finding method		
			a. Fixed-point Iteration			
			b. Newton-Raphson Method			
		10	c. Bracketing Methods		2	
6		11	A5. Arrays	❑ Explain how 1-D arrays are defined and processed in C ❑ Explain how to process strings in C		
			a. 1-D Arrays and Strings			
	3	12	b. Multi-dimensional Arrays	❑ Explain how multi—dimensional arrays are declared and processed ❑ Develop a program that uses two-dimensional arrays to solve problems		
			c. Passing Arguments to Functions	❑ Discuss how arrays are passed as parameters to functions		
7		13	B3. Systems of Linear and Non-linear equations	❑ Compute for an approximate solution to a system of equations using various iterative methods. ❑ Develop a program that approximates the solution to a system of equations using iterative methods.		
			a. Direct Methods+		3	
		14	b. Iterative Methods			1
8	4	15	A6. Pointers	❑ Explain how to use pointers in C ❑ Differentiate passing arguments by value from passing arguments by reference ❑ Develop a program that uses pointers in handling arrays ❑ Demonstrate how to allocate, reallocate, and free memory space using dynamic memory allocation functions		
			a. Addresses, Values, and Pointer Operations			
			b. Passing parameters by-value & by-reference			
		16	c. Dynamic Memory Allocation		4	
9		17	B4. Interpolation, Approximation, & Curve-Fitting	❑ Produce an interpolation or function approximation polynomial for a given set of values. ❑ Derive an interpolating polynominal through a given data points ❑ Develop a program that interpolates or approximates a function from a given set of values.		
			a. Direct Method			
			b. Lagrange and Newton Polynomials			
	5	18	c. Least Squares Method	❑ Implement the Least Squares Method to fit a line or a parabola to a given set of data. ❑ Apply data linearization to fit a given set of data to any nonlinear function.		
			d. Data Linearization			
10		19	B5. Numerical Differentiation and Integration	❑ Compute for an approximate of a derivative of a function using various difference formulas. ❑ Develop a program that approximates derivatives a functions.		
			a. Difference Formulas			
			b. Higher-ordered Derivatives			
		20	c. Newton-Cotes Quadrature Formulas	❑ Apply quadrature formulas to approximate an integral or a function that interpolates a given data set. ❑ Develop a program that impements quadrature formulas.		
			d. Multiple Integrals		5	2
11	6	21	B6. Numerical Solutions of ODEs	❑ Calculate the approximate solution of a first-order ODEs ❑ Approximate the solution of Initial value problems governed by higher-order ODEs and system of ODEs ❑ Develop a function or program that solves inital value problems		
			a. Initial Value Problems			
			i. First-order Equations			
12			ii. Higher-order and System of Equations			
			b. Boundary value problems	❑ Approximate solution of a second-order boundary value problems	6	
13		22	B7. Numerical Solutions to Partial Differential Equations	❑ Define and classify partial differential equations (PDEs). ❑ Discuss how FDM can approximate the solution of PDEs. ❑ Apply FDM to approximate the solution of common PDEs. ❑ Develop a program that solves problems governed by PDEs		
			a. FDM for Hyperbolic and Parabolic PDEs			
14	7	23	b. FDM for Elliptic PDEs: Direct & iterative			
15		24	A7. Structures*	❑ Discuss how to define complex data types using structures ❑ Discuss how to manipulate data stored in structures	7	3

HOW WILL YOU BE GRADED?

As mentioned earlier, non-graded assessment tools – worksheets – are for the purpose of self-evaluation. But please take note that **you are required to submit your output for these assessments** in order to unlock the machine exercises modules in our course page.

Machine Exercises

Machine Exercise (MX) after each of the eight clusters will comprise 60% of your grade. Sixty percent of your final grade will be the **average of these seven MXs**.

Problem Set

Three (3) **Problem Sets** (PS) will be given in the course and will be used to assess your mastery of the topics, especially on numerical methods. Each problem set will consist of two to four problems that you will solve and need to show complete solution including detailed manual computations.

Students may submit solutions to Problem Sets by group. A group may consist of a maximum of three (3) students working on a particular PS. Groupings may vary per Problem Set, i.e, a student may work individually on PS1, then work as a group on PS2, then work with another group on PS3. A student's individual PS grade is computed as

PS-Individual-Grade = PE_Grade x (PS-Group-Score)

where PE_Grade is the average Peer-Evaluation grade that your groupmates give you on that PS.

** Because learning in this course is self-paced and our circumstances during this pandemic are different individually, you may submit your MXs and PS anytime until **January 6, 2022**. It is still highly recommended that you submit the machine exercises on the suggested dates as to not be overwhelmed by the requirements piling up as the semester nears its end.

Class Participation

Class participation is highly encouraged in this class. Students who participate by (i) asking or (ii) answering questions during synchronous or asynchronous sessions will receive a maximum of 4% bonus in Final Grade.

In summary, therefore, your grade will be computed as follows:

Final Grade = 0.60*MX_average + 0.40*PS_average

Your final UP grade in this course will be computed following the table below. No grade of 4.00 will be given in this pass or fail course.

Final Grade	Equivalent Grade	Final Grade	Equivalent Grade
92-100	1.00	72-below 76	2.25
88-below 92	1.25	68-below 72	2.50
84-below 88	1.50	64-below 68	2.75
80-below 84	1.75	60-below 64	3.00
76-below 80	2.00	Below 60	5.00

WHO IS YOUR INSTRUCTOR?

I am **Eric Augustus J. Tingatinga**, a full-time Professor from the Structural Engineering Group in the Institute of Civil Engineering UP Diliman. My research interests include earthquake engineering, nonlinear analysis of buildings, and structural dynamics. I am happily married and have three kids aged 16, 13, and 9 – that's why I don't work during weekends and even at home (during face-to-face setup, of course). I like photography, programming, and weekend getaways.

You may contact me through:

- Email: **eating@up.edu.ph**
- Facebook/Messenger: **fb.com/eat1ng**
- Contact Number: **+639172516374**

My consultation hours are: 9-12Noon TTh, 1-5 PM MW.

You can call me **Sir Eric** (Please don't call me Sir Eating – I hate it when people call me that**)**.

HOW SHOULD YOU CONDUCT YOURSELF?

(House Rules adopted from Padilla, 2020)

You are requested to try hard to observe these house rules in our class:

1. Be kind to yourself and be gentle to others. This is already a challenging time as it is. Please do not make it any more difficult for yourself and/or for others. Try your best to observe the following, but if you cannot do so, please tell me as soon as possible. **Do believe that things have a way of working out eventually.**

    a. Be mindful of time and its importance. Please try hard to
    - visit the course site regularly to be updated on relevant matters;
    - follow the course schedule; and
    - complete/submit requirements on time, following specifications/guidelines.

    b. Do your part in the learning process. Please try hard to
    - read/study assigned materials/resources;
    - participate in class discussions; and
    - treat others with respect and empathy, if not compassion.

    c. Remember the values of UP. Please try hard to show honor and strive for excellence by
    - respecting intellectual property rights (e.g., NOT sharing or distributing any part of the copyrighted materials which have been made available to you solely for the educational purposes of this course);
    - observing academic honesty and the ethics of scholarship in the course;
    - keeping in mind that any form of academic dishonesty (like plagiarism) automatically merits a final grade of 5.0 in the course and may be subject to additional disciplinary measure; and
    - following the Basic Plagiarism Rules below.

      Basic Plagiarism Rules
      You have committed plagiarism when:
      1. You used ideas not your own, and did not cite the source, even if you reworded the text entirely.
      2. You used the wording or ideas (even if reworded) without citing the source, even if you did not intend to plagiarize or did not know you were plagiarizing.
      3. Using at least six words, in succession, of a material without quoting and citing its source.
      4. Using the same words and ideas in another language (translation).
      5. Submitting the same text for two different subjects/teachers/purposes (you can plagiarize yourself).
      6. Patching together, cutting up and pasting words to create a mosaic of words by the same or by another/other author/s.
      7. Patching up together ideas to create a mosaic of ideas by the same or by another/other writer/s.
      8. Misquoting the words of an author.
      9. Wrongly citing bibliographic data of the source, including wrongly attributing text to a source, or inventing a bibliographic source for certain words/ideas.

- from Torrecampo, R.M. (2009). DLL 210 course guide. UP Open University, Laguna

Note: All submissions in this course will go through a plagiarism check.

d. <u>Reach out when you need help</u>. Immediately get in touch with a classmate and/or me for support.

e. <u>Help your classmates and/or the instructor whenever possible</u>. All members of the class – students and instructor – WE – are on this journey together.

2. Follow all specific guidelines for different aspects of the course. This will facilitate the teaching-learning process, which will benefit you.

3. Seize every second – take advantage of every learning moment…but look beyond the moment! Will to learn and enjoy, not simply to earn 3 units and get a passing or good grade. You can do it!

4. If you have a medical or any other condition that could get in the way of your optimum performance in this class, please inform me right away so I can address the matter in a timely and appropriate manner.

5. If, for some justifiable reason and/or unavoidable/unforeseen circumstance, an adjustment in the course needs to be made, please do get involved in the change/revision process that I will facilitate.