**CENTRAL UNIVERSITY**

**SCHOOL OF ENGINEERING AND TECHNOLOGY**

**DEPARTMENT OF COMPUTER SCIENCE & INFORMATION TECHNOLOGY**

**Course Code**: COMP306 **Credit Hour(s)** : **3** **Webpage** :

**Course Title**: Software Engineering

**Course Lecturer: Regina Naa Dedei Crabbe Room: Block C2**

**Email Address:** [**rndcrabbe@gmail.com**](mailto:rndcrabbe@gmail.com) **Tel no.: 0244893514**

**Office Hours: Tuesdays & Thursdays**

# Course Objective

At the end of this course, students should be proficient in the following:

* requirement gathering techniques in software development
* use case modelling tools and techniques
* best practises in the world of software engineering

# Course Description

This course takes an engineering approach to software development. It focuses on the necessary conceptual and analytical tools for systematic and rigorous development of software systems. By the end of the course, students should be able to cover and understand the fundamentals of software engineering, including techniques used in large scale scientific or technical software development, requirement analysis, specification, systems design, implementation, testing, validation, verification and maintenance. Other topics to be discussed includes object-oriented analysis, object oriented design, implementation, testing, tools and techniques for software development such as UML, CASE tools.

# Learning Outcomes

By the end of this course, students should be able to

* clearly define what software engineering is
* model software requirement with appropriate tools
* apply various strategies to build software

# Instructional Methods

Instructional approaches to be used during the course (e.g., lectures, seminars, laboratory activities, tutorials, group projects). Note that attendance is also a requirement.

# Required Course Materials and Readings

Sommerville, I. (2011), *Software engineering: An Engineering Approach.*(9thed.). Michigan, MI: John Wiley.

Pressman, R. & Maxim, B. (2014). *Software Engineering: A Practitioner's Approach*. (8th ed.).New York, NY: McGraw-Hill.

Tsui, F., Karam, O., & Bernal, B. (2022). *Essentials of software engineering*. Jones & Bartlett Learning.

Mohapatra, H., & Rath, A. K. (2020). *Fundamentals of software engineering: designed to provide an insight into the software engineering concepts*. BPB Publications.

Roggenbach, M., Cerone, A., Schlingloff, B. H., Schneider, G., & Shaikh, S. A. (2021). *Formal methods for software engineering*. Springer, Switzerland.

Loubser, N. (2021). *Software Engineering for Absolute Beginners*. Apress.

# Evaluation

Assignment 10%

Presentation 10%

Mid-Semester Exam 20%

End of Semester Exam 60%

Total 100%

# Commit To Academic Integrity

Students in the department are expected to maintain **high degrees of professionalism,** **commitment to active learning, participation and academic integrity every time**.

# Academic Dishonesty

Please note that students involved in academic dishonesty will receive a **ZERO** mark on the particular component in which the infraction occurred and a notation of academic dishonesty in the departmental office. This may also reflect on references written by the department.

**It is the student’s responsibility to understand what constitutes academic dishonesty.**

# Missed Exams / Tests / Assignments

**Assignment Submission**: Assignments must be received on the due date specified for the assignment.

**Lateness Penalty:** Assignments received later than the due date will be penalized Exceptions to the lateness penalty for valid reasons such as illness, etc., may be entertained by the Lecturer but will require supporting documentation (e.g., a doctor’s letter).

**Missed Tests:** Students with a documented reason for missing a course test, such as illness, which is confirmed by supporting documentation (e.g., doctor’s letter) will be handled by the Lecturer.

**WEEK BY WEEK COURSE SCHEDULE / ORGANISER:**

| **Week** | **Topic** | **Activities** | **Due Date** |
| --- | --- | --- | --- |
| 1 | **Software Engineering**   * Definition and principles of software engineering * Software development life cycle (SDLC) models * Role of software engineers and their responsibilities * Importance of software engineering in modern applications * Techniques used in Large Scale Scientific or Technical Software Development | Lectures begin  Assign reading |  |
| 2-3 | **Requirement analysis**   * Understanding and elicitation of software requirements * Requirements gathering techniques (interviews, surveys, workshops, etc.) * Requirements documentation and analysis * Managing requirements changes and scope creep   **System specification**   * Translating requirements into system specifications * Functional and non-functional requirements * Use cases, scenarios, and user stories * Creating clear and unambiguous specifications | Lecture |  |
| 4-5 | **Systems design**   * Overview of software design process * Architectural design and design patterns * Data design and database modeling * User interface design and usability considerations   **System implementation**   * Coding practices and programming languages * Unit testing and integration testing during implementation * Version control and collaborative development * Code reviews and software quality assurance | Lecture/Assign reading |  |
| 6 | **System testing**   * Importance of testing in software development * Types of testing (unit, integration, system, acceptance, etc.) * Test planning and test case development * Test automation and test frameworks | Lecture |  |
| 7 | **Validation and Verification**   * Differentiating between validation and verification * Techniques for validating and verifying software * Software validation with user feedback and real-world scenarios * Ensuring software meets specified requirements | Lecture/Assign reading  MID SEMESTER EXAM |  |
| 8 | Maintenance   * Software maintenance phases and activities * Corrective, adaptive, and perfective maintenance * Handling software updates, bug fixes, and enhancements * Software evolution and managing legacy systems | Lecture/Assign reading |  |
| 9 | **Object-oriented analysis, design, implementation and testing**   * Introduction to object-oriented concepts (classes, objects, inheritance, polymorphism, etc.) * Testing object-oriented software with a focus on encapsulation and modularity | Lecture/Assign reading |  |
| 10-11 | **Introduction to UML design**   * Understanding Unified Modeling Language (UML) * UML diagrams for modeling software systems (use case, class, sequence, etc.) * Use of UML in different phases of software development | Lecture/Exercise |  |
| 12 | **CASE Tools**   * Introduction to Computer-Aided Software Engineering (CASE) tools * Popular CASE tools and their features * Benefits and limitations of using CASE tools in software development * Integrating CASE tools into the software development process. | Lecture/Assign reading |  |
|  |  |  |  |
|  |  | Lecturers end |  |
|  |  | Revision Week |  |
|  |  | Exams begin |  |
|  |  | Exams end / vacation |  |