Effective Teaching of "Advanced Programming"

(A Module in the CSEDU – Certificate Program in CS Education)

This module aims to help teachers in colleges/universities (attendees) improve their teaching of the Advanced Programming course, as per the AICTE syllabus for it (given at the end of this document). I.e. After this module, the attendees will improve their teaching of this course, leading to improved learning by their students and more students achieving the learning outcomes of the AICTE course. This module is part of the Certificate Program in CS Education initiative (csedu.iiitd.ac.in). The main learning outcomes of this module are: (At the end of the module, an attendee will:)

- Have a clearer understanding of the importance of this course in a CSE program, the desired learning outcomes established in the revised AICTE course (attached herewith), and the course syllabus.
- For each topic in the AICTE syllabus, have a deeper understanding of concepts, what is important, how to teach, what type of assignments to give, etc.

Requirements for Module Participants

- Have taught any programming courses at a basic level (any language).
- Have access to a good laptop (or Desktop) and internet connection.
- Have basic knowledge of Java language.
- Commit to spending at least 5 hours per week (avg) for the module.

Module Summary

Each weekly session will discuss the topics listed below, how to effectively teach them and sample assignments. Lab assignments will be given each week. Some sessions will have in-class assignments where the participants will be expected to work out something and discuss it with the instructor/other participants. This course will use the Java programming language, but participants are expected to have basic familiarity with Java.

The week-wise syllabus for this module is as follows:

Wk	Topics	Assignment - Self-work
1	Introduction to object-oriented programming (declaring/defining classes and methods, encapsulation, working with objects)	Install necessary software, gain familiarity with IDE (NetBeans, Eclipse, etc.), write program demonstrating basic OOP concepts (classes/objects), compile and run the program, debug by setting breakpoints
2	Class relationships (association, composition, objects as parameters)	Take home programming assignment based on identifying different class relationships and implementing them.
3	Interfaces	Take home assignment based on declaring

		and implementing interfaces
4	Polymorphism using interfaces	Take home assignment based on understanding method resolution/polymorphism using interfaces, understanding declared v/s actual type
5	Inheritance	Take home assignment covering code reuse, IS-A relationship
6	Polymorphism using inheritance	Take home assignment based on method overloading and method overriding.
7	Defensive programming and exception handling	Take home assignment based on handling basic exceptions (both checked and unchecked type), writing and handling custom exceptions
8	Unit testing	Take home assignment on JUnit
9	Basic modelling technique (class diagrams and use case diagrams)	Apply the learnings to model some of the programming assignments used in this course
10	Design patterns	Take home assignment based on identifying different design patterns and implementing them.
11	Multithreading and thread pools	Take home assignment based on parallelizing some simple algorithm both by using explicit multithreading, and thread pool runtime
12	Event-driven programming using JavaFX	Take home assignment on creating Graphical User Interfaces (GUIs)
13	Tools for plagiarism detection	Usage

Schedule

The module will meet once a week. In addition, a weekly help/lab session to clear doubts and help with the assignment will also be provided through TAs. Details about joining these sessions will be provided later.

- Weekly session: Monday 4.30 PM - 6 PM
- Weekly lab/help session: Friday 4.30 PM - 6 PM

Text to be used for the Module

We will not follow a specific textbook, but here are some recommended books/resources for reference:

Grady Booch, Robert A. Maksimchuk, Michael W. Engle, Bobbi J. Young, Jim Conallen, Kelli A. Houston. Object-Oriented Analysis and Design with Applications

Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides, and Grady Booch. Design Patterns: Elements of Reusable Object-Oriented Software

Resources to be provided to attendees

- Lecture Notes / ppt for the different topics in the course
- Some sample assignments for each of the major modules
- Some online resources that the attendees can use to revise/update their knowledge