M.Sc.CSIT
First Semester
Syllabus

Object Oriented Software Engineering

Course Title: Object Oriented Software Engineering
Course No: CSIT. 512
Nature of the Course: Theory + Lab
Semester: I

Full Marks: 60+40
Pass Marks: 30+20
Credit Hrs: 3

Course Description:

Object-Oriented Software Development is an approach/paradigm of developing software by identifying and implementing a set of objects and their interactions to meet the desired objectives. This course introduces the object oriented software engineering; it will enable the learner how to use the OO technology efficiently in software design and development process.

Course Objectives:

- To display understanding and the ability to apply object-oriented programming principles.
- To apply skills relevant for academic progression and career development within the sector.
- To explore and analyze different analysis and design models, such OO Models and Structured Analysis and Design Models.
- To show an ability to use the graphical UML representation using tools.

Course Contents

Unit 1: Introduction (6 Hrs.)

1.1. Software life cycle models, Requirement analysis and specification, object oriented software development.

Unit 2: Object orientation (8 Hrs.)

2.1. Classes and objects, Instance variables Methods, operations and polymorphism, Organizing classes into inheritance hierarchies, The effect of inheritance hierarchies on polymorphism and variable declarations

2.2. Concepts that define object orientation, Function/ Data methods, Object –oriented analysis, Object –oriented construction, Object –oriented testing

2.3. A program for manipulating postal codes, Classes for representing geometric points, Measuring the quality and complexity of a program ,Difficulties and risks in programming language choice and OO programming

Unit 3: Architecture and design (8 Hrs)

3.1. Architecture-model architecture, requirements, analysis, design, implementation and test model, analysis, construction, Real-time-classification of real time systems

3.2. Database-RDBMS, Object DBMS, Component based software engineering, component management, testing-on testing, Unit, integration system and the testing process.

Unit 4: Project management

4.1. Managing object-oriented software engineering, Project selection and preparation, product development organization

4.2. Project organization and management, project staffing, software quality assurance, software metrics, agile project management activities.

Unit 5: Modern techniques of object oriented analysis and design (8 Hrs)

5.1. Object oriented analysis and design, hierarchical object-Oriented design, object Modeling technique and responsibility-driven design

Case Studies (15 Hrs.)

6.1. Case studies related to object oriented software Engineering

References

2. Ivar Jacobson-Objected software engineering.
3. Ian Sommerville-Software Engineering
4. Grady Booch-Object-Oriented analysis and design
5. Bernd Bruegge & Allen H. Dutoit, Object-Oriented Software Engineering Using UML, Patterns, and Java™
6. Timothy C. Lethbridge Robert Laganière , Object-Oriented Software Engineering

Python for Data Science

Course Title: Python for Data Science
Course No: CSIT. 514
Nature of the Course: Theory+Lab
Semester: I

Full Marks: 60+40
Pass Marks: 30+20
Credit Hrs: 3

Course Description:
The fundamentals of procedural and object-oriented Python programming are covered in this course, along with a thorough examination of the use of various libraries and their applications in machine learning and data science.

Course Objectives:
- To provide the ability to write Python programs using both procedural and object-oriented approaches.
- To make it possible for students to use the commonly-used Python libraries.
- To apply python programming and libraries in implementing machine learning algorithms and models.

Unit 1: Procedural Programming (12 Hrs.)

1.1. Keywords, Identifier, Data types, variables, Literals, Operators, Operator Precedence, Escape sequences, Comments.
1.2. Ternary operator, Conditional statements, Looping Statements, Jump Statements.
1.3. Functions, Passing Arguments, Passing Variable Number of Arguments, Returning values, Returning Objects, Returning Multiple Values, Global and Local variables, Global Keyword, Recursive functions, Functional Tools.
1.4. String Functions, String Concatenation, String operations, String slicing, formatted strings.
1.5. Lists, Tuples, Sets, and Dictionaries. Functions, methods, and operations of each data structure, List Comprehension.

Unit 2: Object Oriented Programming (10 Hrs.)

2.1. Enumerations, Zip and Argument Unpacking, Exception Handling, File Handling, Regular Expressions.
2.2. Class, Object, constructors, access modifiers, instance and class variable, static methods and class methods, method overloading, operator overloading.
2.3. Inheritance, implementing different types of inheritance, method overriding, abstract classes.

Unit 3: NumPy, Pandas, and Matplotlib (10 Hrs.)

3.1. NumPy Basics, Array and vectorized processing, operations between arrays and scalars, slicing and indexing, multi-dimensional array, Universal Functions, data processing with arrays, array object, array functions, File input and output with arrays, Linear Algebra with arrays, random number generation.
3.2. Pandas Data structure, Essential Functionalities, Summarizing and Computing Descriptive Statistics, Handling Missing Data, Hierarchical Indexing.
3.3. Introduction, Plotting Functions in pandas, Plotting Maps, Python Visualization Tool Ecosystem.

Unit 4: Data Preprocessing in Data Science (13 Hrs.)

4.1. Reading and Writing Data in Text Format, Binary Data Formats, Interacting with HTML and Web APIs, Interacting with Databases.

4.2. Combining and Merging Data Sets, Reshaping and Pivoting, Data Transformation, String Manipulation.

4.3. GroupBy Mechanics, Data Aggregation, Group-wise Operations and Transformations, Pivot Tables and Cross-Tabulation.

Laboratory Works:

Students are required to write procedural and object-oriented Python programs. Additionally, they must leverage several libraries that have been covered in class and find solutions to a variety of problems involving data preprocessing and data transformation.

Text Book:

1. Johannes Ernesti and Peter Kaiser, Python 3: The Comprehensive Guide, SAP Press, First Edition, 2022.
2. Sridhar, Python Programming, Pearson Education, First Edition, 2023.
3. William McKinney, Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython, O'Reilly Media, 2nd Edition, 2017.

Reference Books:

1. Daniel Zingaro, Learn to Code by Solving Problems: A Python Programming Primer, No Starch Press, First Edition, 2021
2. Codeone Publishing, Python Programming for Beginners: The #1 Python Programming Crash Course to Learn Python Coding Well and Fast (wi
3. th Hands-On Exercises), First Edition, 2022.

Advanced Operating Systems

Course Title: Advanced Operating Systems
Course No: CSIT. 511
Nature of the Course: Theory + Lab
Semester: I

Full Marks: 60+40
Pass Marks: 30+20
Credit Hrs: 3

Course Description:

Major algorithms used in process management, memory and storage management, protection and security, distributed systems, real-time systems, and multimedia systems are discussed and demonstrated in this course.

Course Objectives:

- To demonstrate algorithms used in process, deadlock, memory and disk management.
- To provide exposure to current state-of-art research in operating system
- To provide concepts of distributed operating system, real-time operating system and multimedia systems.

Course Contents

Unit 1: Process Management (13 Hrs)

1.1. **Process management:** Process Model, Modeling Multiprogramming, Critical Section, Race Condition.
1.2. **Inter-process communication:** Concepts and Problems of Busy Waiting Techniques, sleep and wakeup, semaphore, mutex, monitor,.
1.3. **Process scheduling:** Round robin, Issues with quantum size, Research Article Based Study on at least two variants of *Round Robin with variable quantum size.*
1.4. **Deadlocks:** Deadlock prevention, avoidance, detection, and recovery, *Paper Based study on Deadlock Recovery Technique.*

Unit 2: Memory Management (13 hrs)

2.1. **Paging:** Page tables, speeding up page tables, Page tables for large memories.
2.2. **Page replacement algorithms:** Brief overview of LRU and clock, *Research Papers on LRFU and LIRS and CFLRU,* Design Issues for paging, Segmentation, Segmentation with paging.

2.3. Storage Management: Implementing files and directories, log structured file system, journaling file system, Disk space management, *flash file systems, high performance flash disks.*

2.4. Disk scheduling algorithms: Disk Formatting, Error Handling, *Paper Based study on Disk scheduling algorithms.*

Unit 3: Protection and Security (6 Hrs)

3.1. System Protection: protection principles and domain, access matrix and its implementation, access controls and rights, capability-based system, language-based protection.

3.2. System Security: Program threats, system and network threats, Cryptography as a Security Tool, User Authentication, Firewalling, Computer security classification, Software Attacks, *Paper based study on Capabilities & access control and exploitation techniques and malware.*

Unit 4: Distributed and Special Purpose Systems (13 Hrs.)

4.1. Distributed Operating Systems: Types of Network based Operating Systems, Network structure and topology, Communication structure and protocols, Robustness, Design Issues,

4.2. Distributed File Systems: Naming and Transparency, Remote File Access, Stateful versus Stateless Service, File Replication.

4.3. Real-Time Systems: Real time system characteristics and kernel features, Implementing real-time operating system, real-time CPU scheduling.

4.4. Multimedia Systems: Multimedia system overview, multimedia kernels, compression algorithm, CPU scheduling, disk scheduling.

Laboratory Works

The only way to learn this course effectively is to put a strong emphasis on the implementations of the algorithms and techniques covered in class. Since algorithms can be simulated and analyzed using any high level language, students should be able to do so. Student should simulate for this aim.

- CPU scheduling algorithms
- Page replacement algorithms
- Disk scheduling algorithms
- Deadlock detection and recovery algorithms
- File System implementation techniques

- Security techniques

Recommended Book

1. Andrew S. Tanenbaum, Moderm Operating Systems, PHI.
2. Silberschatz, Galvin and Gagne. Operating System Concepts, John-Wiley.

Research Methodology for Computer Science

Title: Research Methodology for Computer Science
Course No: CSIT.515
Nature of the Course: Theory + Lab

Full Marks: 60 + 40
Pass Marks: 30 + 20
Credit Hrs: 3

Course Description:
This course is designed to provide students with a comprehensive understanding of concepts of research methodology. The course covers various aspects of research methodology including fundamental concepts, research design, data collection, literature survey, research models, technical writing and publication, plagiarism, intellectual rights, and ethics.

Course Objectives:
The course aims to achieve the following objectives:

- Introduce concepts of research methodology
- Provide concepts of research design
- Familiarize data collection and analysis strategies
- Provide concepts of data collection and data analysis
- Familiarize with literature survey and writing problem statements
- Present research methods for science
- Perform technical writing and publishing activities
- Acquaint with plagiarism, intellectual rights and ethics in research

Unit I: Introduction (5 Hrs.)
Foundations of Research(Meaning, Objectives, Motivation, Utility), Types of Research, Research Approaches, Significance of Research, Scientific Method, Research Process, Research Problem, Research Question, Hypothesis

Unit II: Research Design (5 Hrs.)
Research Design, Need for Research Design, Features of a Good Research Design, Exploratory Research Design, Descriptive Research Designs, Experimental Design

Unit III: Data Collection and Analysis (8 Hrs.)
Sampling, Scales of Measurement, Collection of Data, Observation, Surveys, Errors, Analysis of Data, Descriptive Statistics, Inferential Statistics, Statistical Tests: Z-test, T-test, ANOVA, Chi-Square; Statistical Packages: SPSS, SAS, Stata, Monitoring Research

Unit IV: Literature Survey and Problem Statement (8 Hrs.)

Conducting Background Research, Resources for Literature Survey, Reading Scientific and White Papers, Research Reports, Summarizing Findings of Literature Survey, Formulation of Problem Statement, Scope and Significance of Research Problem

Unit V: Research Models for Science (5 Hrs.)
Algorithmic Research: Analysis of Algorithm, Design of Algorithms, Methods of Scientific Research, Modeling: Steps in Modeling, Research Models, Simulations: Types of Simulation Models, Tools for Simulations

Unit VI: Technical Writing and Publishing (7 Hrs.)
Free Writing, Technical Writing, Writing Strategies, Contents of Research Papers, Reports, Theses, Review System, Selection of Journal, Identifiers to identify Journals and Research Papers, Subscription Based and Open Access Journals, Journal Ranking and Journal Metrics, Ranked Journals of Computer Science, Citation Index, Databases, Search Engines, Author Identifiers, Documentation Tools, Bibliography Tools, Presentation Tools,

Unit VII: Plagiarism, Intellectual Rights and Ethics (7 Hrs.)
Plagiarism, Types of Plagiarism, Plagiarism Policies, Techniques, Avoiding Plagiarism, Plagiarism Checking, Intellectual Property, Forms of Intellectual Property Rights, Role of Intellectual Property Rights, Ethics in Research, Concepts in Ethics, Fraud and Misconduct in Science, Ethics in Science Practical, Whistle Blowing, Scientific Values and Code of Conduct

Laboratory Works:

Laboratory works include implementing and simulating the concepts in above mentioned chapters using appropriate platforms and tools.

Text/Reference Books

1. Research Methodology: A Practical and Scientific Approach, Vinayak Bairagi and Mousami V. Munot, CRC Press
2. Research Methodology and Scientific Writing, C. George Thomas, Springer
3. Engineering Research Methodology: A Practical Insight for Researchers, Dipankar Deb, Rajeeb Dey, Valentina E. Balas, Springer
4. Research Methodology, Vinod Chandra, Anand Harindran, Pearson Education India
5. Research Methodology: Methods and Techniques, C. R. Kothari, New Age International Publishers
6. Research Methodology in Computer Science, Ryhan Ebad, Centrum Press

Computational Complexity

Course Title: computational complexity
Course No: CSIT.513
Nature of the Course: Theory+Lab
Semester: I

Full Marks: 60+40
Pass Marks: 30+20
Credit Hrs.: 3

Course Description:
The main complexity classes and the theory of computational complexity are introduced in this course. The ability to categorize computational problems based on how challenging they are to solve is one of the most significant discoveries to come out of theoretical computer science. This classification has demonstrated that many computational issues are hard to solve in a reasonable length of time, and that many more are impossible to solve altogether. One requires a rigorous model of computation and a technique to compare problems of various types in order to categorize problems in this way. This course presents these concepts and demonstrates their use.

Course Objectives:

- To classify decision problems into appropriate complexity classes, including P, NP, PSPACE and complexity classes based on randomized machine models and use this information effectively.
- To state precisely what it means to reduce one problem to another, and construct reductions for simple examples.
- To classify optimization problems into appropriate approximation complexity classes and use this information effectively.
- To use the concept of interactive proofs in the analysis of optimization problems.

Unit 1: Models of computation

1.1. Turing machine (Computation, Complexity)

1.2. Asymptotic notations (Big oh, Big omega, Big theta, little oh, little omega and little theta)

1.3. Turing machine variants, Universal Turing machine, Random Access Machine, Extended Church-Turing Thesis.

1.4. Time and Space Complexity classes.

Unit 2 Computational Complexity Theory

2.1. Basic Concepts: Complexity Theory, Complexity Classes: P, NP, NP-Hard and NPComplete, Decision Problems and Language Recognition Problems

2.2. Problem Reduction: Reduction, Polynomial time reduction, Cooks Theorem, Proving NP-Completeness: Formula Satisfiablity, 3SAT, CLIQUE, Vertex Cover, Hamiltonian Cycle, TSP, SubsetSum.

2.3. NP-Hard Code Generation Problems: Code Generation with Common Subexpression, Implementing Parallel Assignment Instructions.

2.4. Coping with NP-Completeness: Performance ratios for approximation algorithms, Approximated Algorithms: Vertex Cover, TSP, Set Covering, Subset Sum.

Unit 3: Diagonalization and Relativization

3.1. Undecidability of halting problem, Hierarchy theorem (space hierarchy theorem and time hierarchy theorem)

3.2. Hierarchy theorem for non-determinism, Lander's theorem

3.3. Relativization, The Baker Gills-Solovay Theorem, The oracle polynomial time hierarchy

Unit 4: Space Complexity

4.1. Space and time, PSPACE and TQBF, Savitch's theorem, Immerman-szelepcsenyi theorem

4.2. Complete problem with respect to log-space reduction, Consequence of NL or P-completeness, Complete problem for NL, Complete problem for P, AL=P

4.3. Circuit Complexity, polynomial size circuit (P/poly,information-theoretic bound, the Karp-Lipton theorem)

4.4. Uniformity, Bounded depth circuit

4.5. Descriptive complexity (first order logic, bit string, directed graph, second order logic, counting with first order logic, Fagin's theorem)

Unit 5 Randomizes classes and Pseudo-randomness

5.1. One sided error (RP, coRP, ZPP), $P \subset RP \subset NP$, a problem in coRP: Polynomial identity testing, Las Vegas algorithm and ZPP

5.2. Two sided error: BPP, Adleman's theorem: $BPP \subset P/poly$, randomized log space algorithm.

5.3. Complete problem for BPP, Practical implementation of BPP, derandomizing BPP, secure pseudorandom generator, consequence for P=BPP, Hardness.

Laboratory Works:

Student should implement various algorithms discussed in the class and should perform empirical analysis of algorithms having similar complexity.

Recommended Book:

- S. Arora and B. Barak. *Computational complexity: A modern approach.* Princeton University 2007
- Christos H. Papadimitriou. Computational complexity. Addison-Wesley, 1994
- Ding-Zhu Du, Ker-I Ko. Theory of Computational Complexity. John Wiley & Sons, 201.

Seminar I

Course Title: Seminar I
Course No: CSIT.516
Nature of the Course: Seminar
Semester: I

Full Marks: 25
Pass Marks: 12.5
Credit Hrs.: 1

Course Description:
This one credit course is meant to give students practice speaking in front of a scientific audience and to explore topics in detail. Students will research topics and organize presentations for faculty members and other students. The topics may be any aspect of the computer science and information technology and must be approved by the specified advisor in advance.

Course Objectives:
- To enhance skill of communication and presentation.
- To enable students to face audience confidently.
- To encourage habits of self-learning.
- To enhance skills of academic writing.

Seminar Report Format:

Cover Page

Letter of Approval

Abstract

Table of Contents

Chapter 1: Introduction

1.1. Introduction

1.2. Problem Statement

1.3. Objectives

Chapter 2: Background and Literature Review

2.1. Background

2.2. Related Works

Chapter 3: Methodology

3.1. Methodology

3.2. Tools Used

3.3. Experimental Environment

3.4. Performance Measures

Chapter 4: Results and Discussion

4.1. Experimental Results

4.2. Result Analysis and Interpretation

Chapter 5: Conclusion and Recommendation

5.1. Conclusion

5.2. Recommendations

References

Referencing and citation should be done in IEEE format.