NP-Completeness: **(10%)**
Polynomial time, polynomial time verification, NP-completeness and reducibility

Approximation algorithms: **(10%)**
The vertex cover problem, Traveling salesman problem, the set-covering problem

Main Reading

1. Cormen Thomas, L. Charles, R. Ronald, S. Clifford, "Introduction to Algorithms", Second Edition, EEE, PHI.

Supplementary Reading

1. Algorithm Design by Jon Kleinberg and Eva Tardos, Pearson 2006
2. Introduction to the Design & Analysis of Algorithms by Anany Levitin
3. Intoroduction to Algorithms by Udi Manber
4. Algorithms in C++ by Sedgewick
5. Algorithmics the spirit of computing by David Harel 2. Knuth Donald, "The Art of Computer Programming vol I, II, III

CS305 Object Oriented Technology

Prerequisites: CS101, CS201

Course contents:

Object Orientation:

Basic OO Concepts and principles **(15%)**

Class The role of classes as modules and types, uniform type system. The OO style of computation;

Object: definition, basic form, object references, object identity, declaring references, self-reference, run-time object structure, Object creation, References and calls. Operations on references, Object cloning and copying, deep cloning. Composite objects and expanded types. Dynamic aliasing. Class level operations- static methods. Package structure and importing;

Inheritance, Polymorphism, Typing & Binding: inheritance concepts & rules, Deferred features and classes: deferring and effecting a feature, redeclaration, deferred classes, using the original version in a redefinition. The meaning of inheritance: module view and type view. Extension-specialization paradox. The role of deferred class. Multiple Inheritance, feature renaming, approaches to Multiple inheritance in OOP languages, multiple interface inheritance, repeated inheritance, inheritance and assertions, global inheritance-Ex. Class Object in Java, frozen features, assignment attempt, typing and redeclaration, anchored declaration, inheritance and information hiding. Typing: typing problem. Static and dynamic typing. Why static typing. Binding. Covariance and descendent hiding. Contravariance. Advanced Inheritance mechanisms: inheritance versus composition, Inheritance taxonomy

OOP features: **(25%)**

Memory management: modes of object management, space reclamation, detachment, unreachable objects, memory management in object-oriented model, approaches: casual, programmer controlled de-allocation, automatic memory management. Algorithms: reference counting, garbage collection, requirements, case studies; Collection Framework: Use of collection framework; Genericity: horizontal and vertical type generalization,

the need for type parameterization, generic classes, defining and using generic classes, type checking and rules, operations of generic type entities, cost of genericity. Generic programming in C++, Java Collections and Generics. Exception handling: principles, exception objects, language mechanisms in Java, C++, exception nesting, hierarchies of exceptions. I/O & file handling: streams, i/o streams input of built-in types, unformatted input, input of user defined type, ostream output of built-in types, output of user defined types, formatting of output, stream state, I /O exception, file streams, string streams, stream buffers. Concurrency: Threads, multithreading, thread synchronization; Reflection: run time type interrogation, how reflection works in Java/C++; Persistence: persistence closure principle, schema evolution, serialization in Java, using relational databases from OOP like Java, C++. OO databases.

OOAD using UML

OOAD: (5%)

Brief review of OO concepts. Assigning responsibilities. What is analysis and Design? What is OOAD?

UML: (5%)

Introduction.. Main UML diagrams- class diagram, sequence diagram, activity diagram, use case diagram.

Use case modeling: (5%)

Use case diagram, use case descriptions, use case realization using sequence and activity diagrams. Generalization, includes, extends

Analysis/Domain/conceptual model: (20%)

Concepts, attributes, operations. Aggregation, composition and containment. Analysis to Design: Use case to analysis model using boundary, control and entity stereotypes**,** CRC technique

Design Model: (20%)

Design class diagram, visibility, delegation versus inheritance. Association class, qualified associations, reflexive associations, ordered associations. Sequence diagram, activity diagram, state chart diagram, deployment diagram.

Brief introduction to other UML 2.0 diagrams (5%)

Main Reading

1. Timothy Budd, An Introduction to Object Oriented Programming, 3rd Edition, Pearson Education
2. Khalid Mughal, A Programmer's Guide to Java Programming Certification, 3rd Edition, Pearson Education
3. Bertrand Meyer: Object Oriented Software Construction, second edition, PTR PrenticeHall Pearson.
4. Stanley Lippman, C++ Primer, 3rd Edition, Addison Wesley
5. Ken Arnold, James Gosling, David Holmes, The Java Programming Language, Fourth Edition, Addison-Wesley Professional, 2005
6. Bjarne Stroustroup, The C++ Programming Language by, 4th Edition, Addison Wesley
7. Martin Fowler, UML Distilled, 2nd Edition, Addison Wesley, 2003
8. Doug Rosenberg, Matt Stephens, Use Case Driven Object Modeling with UML: Theory and Practice., Apress.

Supplementary Reading

1. Grady Booch, James Rumbaugh, Ivar Jacobson, The Unified Modeling Language User Guide, Addison – Wesley Longman, 1999, ISBN 0-201-57 168 –4.
2. Rebecca Wirfs Brook, Designing Object Oriented Software, PHI
3. Ira Pohl, Object-Oriented Programming Using C++, Second Edition, Addison Wesley