TATA CONSULTANCY SERVICES
Experience certainty.

B.E. /B.Tech in Computer Science & Business Systems

Semester 3

TCS

Computer Science & Business Systems

Semester 3 Curriculum

B.E. /B.Tech in Computer Science & Business Systems

Semester 3

FORMAL LANGUAGE & AUTOMATA THEORY (PCC-CS502)

Introduction: Alphabet, languages and grammars, productions and derivation, Chomsky hierarchy of languages.

Regular languages and finite automata: Regular expressions and languages, deterministic finite automata (DFA) and equivalence with regular expressions,nondeterministic finite automata (NFA) and equivalence with DFA, regular grammars andequivalence with finite automata, properties of regular languages, *Kleene's theorem,* pumping lemma forregular languages, *Myhill-Nerode theorem and its uses,* minimization of finite automata.

Context-free languages and pushdown automata: Context-free grammars (CFG) and languages (CFL), Chomsky and Greibachnormal forms, nondeterministic pushdown automata (PDA) and equivalence with CFG,parse trees, ambiguity in CFG, pumping lemma for context-free languages, deterministicpushdown automata, closure properties of CFLs.

Context-sensitive languages: Context-sensitive grammars (CSG) and languages, linear bounded automata andequivalence with CSG.

Turing machines: The basic model for Turing machines (TM), Turingrecognizable(recursively enumerable) and Turing-decidable (recursive) languages and theirclosure properties, variants of Turing machines, nondeterministic TMs and equivalence withdeterministic TMs, unrestricted grammars and equivalence with Turing machines, TMsas enumerators.

Undecidability: Church-Turing thesis, universal Turing machine, theuniversal and diagonalization languages, reduction between languages and Rice s theorem,undecidable problems about languages.

Basic Introduction to Complexity: Introductory ideas on Time complexity of deterministic and nondeterministic Turing machines, P and NP, NP- completeness, Cook's Theorem, other NP - Complete problems.

Laboratory

1. *YACC, the parser-generating tool(Chapter 5 of Introduction to Automata Theory, Languages, and Computation* John E. Hopcroft, Rajeev Motwani and Jeffrey D. Ullman.*)*

Text Books:

2. *Introduction to Automata Theory, Languages, and Computation* John E. Hopcroft, Rajeev Motwani and Jeffrey D. Ullman.

Reference Books:

1. *Elements of the Theory of Computation,* Harry R. Lewis and Christos H. Papadimitriou.

B.E. /B.Tech in Computer Science & Business Systems

Semester 3

2. *Automata and Computability*, Dexter C. Kozen.
3. *Introduction to the Theory of Computation*, Michael Sipser.
4. *Introduction to Languages and the Theory of Computation*, John Martin.
5. *Computers and Intractability: A Guide to the Theory of NP Completeness*, M. R. Garey and D. S. Johnson.

B.E. /B.Tech in Computer Science & Business Systems

Semester 3

COMPUTER ORGANIZATION & ARCHITECHTURE (PCC-CS 402)

Revision of basics in Boolean logic and Combinational/Sequential Circuits.

Functional blocks of a computer: CPU, memory, input-output subsystems, control unit.

Instruction set architecture of a CPU: Registers, instruction execution cycle, RTL interpretation of instructions, addressing modes, instruction set. Outlining instruction sets of some common CPUs.

Data representation: Signed number representation, fixed and floating point representations, character representation.

Computer arithmetic: Integer addition and subtraction, ripple carry adder, carry look-ahead adder, etc. multiplication – shift-and-add, Booth multiplier, carry save multiplier, etc. Division restoring and non-restoring techniques, floating point arithmetic, IEEE 754 format.

Introduction to x86 architecture.

CPU control unit design: Hardwired and micro-programmed design approaches, design of a simple hypothetical CPU.

Memory system design: Semiconductor memory technologies, memory organization.

Peripheral devices and their characteristics: Input-output subsystems, I/O device interface, I/O transfers – program controlled, interrupt driven and DMA, privileged and non-privileged instructions, software interrupts and exceptions. Programs and processes – role of interrupts in process state transitions, I/O device interfaces – SCII, USB

Pipelining: Basic concepts of pipelining, throughput and speedup, pipeline hazards.

Parallel Processors: Introduction to parallel processors, Concurrent access to memory and cache coherency.

Memory organization: Memory interleaving, concept of hierarchical memory organization, cache memory, cache size vs. block size, mapping functions, replacement algorithms, write policies.

Lab:

1. Circuits on breadboard or simulators

(a) Implementation of Combinational Digital/Boolean Circuits: Adder, Subtractor, Multiplication Module, Division Module, Multiplexer, Demultiplexer, Encoder, Decoder.

(b) Implementation of Sequential Circuits: Counters, Linear Feedback Shift Registers (LFSR)

2. C/C++ programming to understand the formats of char, int, float, double, long etc.

3. Machine language programming on x86 or higher version kits or simulators:

B.E. /B.Tech in Computer Science & Business Systems

Semester 3

(i) Add/subtract/multiplication/division/GCD/LCM

(ii) Accessing some specific memory locations/ports

(iii) Counting odd and even integers from a series of memory locations

(iv) Printing values of selected registers

(v) Handing interrupts

Text Books:

1. *Computer System Architecture* M. M. Mano:, 3rd ed., Prentice Hall of India, New Delhi, 1993.
2. *Computer Organization and Design: The Hardware/Software Interface*, David A. Patterson and John L. Hennessy.
3. *Computer Organization and Embedded Systems*, Carl Hamacher.

Reference Books:

1. *Computer Architecture and Organization*, John P. Hayes.
2. *Computer Organization and Architecture: Designing for Performance*, William Stallings.
3. *Computer System Design and Architecture*, Vincent P. Heuring and Harry F. Jordan.

B.E. /B.Tech in Computer Science & Business Systems

Semester 3

OBJECT ORIENTED PROGRAMMING (PCC-CS503) + Lab

Procedural programming, An Overview of C: Types Operator and Expressions, Scope and Lifetime, Constants, Pointers, Arrays, and References, Control Flow, Functions and Program Structure, Namespaces, error handling, Input and Output (*C*-way), Library Functions (*string*, *math*, *stdlib*), Command line arguments, Pre-processor directive

Some difference between C and C++: Single line comments, Local variable declaration within function scope, function declaration, function overloading, stronger type checking, Reference variable, parameter passing – value vs reference, passing pointer by value or reference, ~~#define constant vs const~~, Operator new and delete, the typecasting operator,Inline Functions in contrast to macro, default arguments

The Fundamentals of Object Oriented Programming: Necessity for OOP, Data Hiding, Data Abstraction, Encapsulation, Procedural Abstraction, Class and Object.

More extensions to C in C++ to provide OOP Facilities: Scope of Class and Scope Resolution Operator, Member Function of a Class, private, protected and public Access Specifier, this Keyword, Constructors and Destructors, friend class, error handling (exception)

Essentials of Object Oriented Programming: Operator overloading, Inheritance – Single and Multiple, Class Hierarchy, Pointers to Objects, Assignment of an Object to another Object, Polymorphism through dynamic binding, Virtual Functions, Overloading, overriding and hiding, Error Handling

Generic Programming: Template concept, class template, function template, template specialization

Input and Output: Streams, Files, Library functions, formatted output

Object Oriented Design and Modelling: UML concept, Use case for requirement capturing, Class diagram, Activity diagram and Sequence Diagram for design, Corresponding C++ code from design

Laboratory

1. Parameter passing: passing parameter by value vs by reference, passing array as constant pointer
2. Function overloading: writing string operations like strcat and strncat, strcpy and strncpy as overloaded functions.
3. Dynamically allocating space for a pointer depending on input and doing this repeatedly, depending on different inputs and finally de-allocating the pointer.

B.E. /B.Tech in Computer Science & Business Systems

Semester 3

4. Define class complex with all possible operations: constructor, destructor, copy constructor, assignment operator with the data members stored as pointer to integers.
5. Define class vector of integers with all possible operations like constructor, destructor, copy constructor and assignment operators
6. Define class matrix of integers with all possible operations like constructor, destructor, copy constructor and assignment operators
7. Define class matrix of integers using vector, with all possible operations like constructor, destructor, copy constructor and assignment operators
8. Define class stack, queue, linked-list, array, set using some data-type (int) with data members kept as private and functions kept in both protected and public sections.
9. Define class complex with all possible operators: constructor, destructor, copy constructor, assignment operator and operators >, <, >=, <=, ==, ++ (pre and post), +, +=, ( ), with the data members stored as pointer to integers.
10. Define class vector of integers with all possible operations like constructor, destructor, copy constructor and assignment operators>, <, >=, <=, ==, ++ (pre and post), +, +=, ( )
11. Define class matrix of integers with all possible operations like constructor, destructor, copy constructor and assignment operators>, <, >=, <=, ==, ++ (pre and post), +, +=, ( ).
12. Define class matrix of integers using vector, with all possible operations like constructor, destructor, copy constructor and assignment operators>, <, >=, <=, ==, ++ (pre and post), +, +=, ( ).
13. Define stack and queue inherited from array class, with standard functions and operators
14. Define a class called ‘array’ with data type passed as template type with constructor, destructor, copy constructor and assignment operators and index operator.
15. Define template functions for compare and use it in the algorithms like bubble sort, insertion sort, merge sort.
16. Formatted input-output examples
17. Input manipulators
18. Overriding operators <<, >>
19. Define class model for complex number, student class, book class and show it using UML diagram as well as concrete class.
20. Show behavioural modelling through sequence diagram and activity diagram for workflow in a typical log-in, log-out situation.

Text Books:

1. *The C++ Programming Language*, Bjarne Stroustrup, Addison Wesley.
2. *C++ and Object-Oriented Programming Paradigm*, Debasish Jana, PHI Learning Pvt. Ltd.

Reference Books:

1. *Programming – Principles and Practice Using C++*, Bjarne Stroustrup, Addison Wesley.
2. *The Design and Evolution of C++*, Bjarne Stroustrup, Addison Wesley.

B.E. /B.Tech in Computer Science & Business Systems

Semester 3

COMPUTATIONAL STATISTICS + Lab

Multivariate Normal Distribution: Multivariate Normal Distribution Functions, Conditional Distribution and its relation to regression model, Estimation of parameters.

Discriminant Analysis: Statistical background, linear discriminant function analysis, Estimating linear discriminant functions and their properties.

Principal Component Analysis: Principal components, Algorithm for conducting principal component analysis, deciding on how many principal components to retain, H-plot.

Factor Analysis: Factor analysis model, Extracting common factors, determining number of factors, Transformation of factor analysis solutions, Factor scores.

Clustering: Introduction, Types of clustering, Correlations and distances, clustering by partitioning methods, hierarchical clustering, overlapping clustering, K-Means Clustering-Profiling and Interpreting Clusters

Laboratory

Python Concepts, Data Structures, Classes: Interpreter, Program Execution, Statements, Expressions, Flow Controls, Functions, Numeric Types, Sequences and Class Definition, Constructors, Text & Binary Files - Reading and Writing

Data Wrangling: Combining and Merging Datasets, Reshaping and Pivoting, Data Transformation, String Manipulation, Regular Expressions

Data Aggregation, Group Operations, Time series: GoupBy Mechanics, Data Aggregation, Groupwise Operations and Transformations, Pivot Tables and Cross Tabulations, Time Series Basics, Data Ranges, Frequencies and Shifting

Visualization in Python: Matplotlib package, Plotting Graphs, Controlling Graph, Adding Text, More Graph Types, Getting and setting values, Patches

Text Books:

1. *An Introduction to Multivariate Statistical Analysis,* T.W. Anderson.
2. *Applied Multivariate Data Analysis, Vol I & II,* J.D. Jobson.
3. *Statistical Tests for Multivariate Analysis,* H. Kris.
4. *Programming Python,* Mark Lutz.
5. *Python 3 for Absolute Beginners,* Tim Hall and J-P Stacey.

Beginning Python: From Novice to Professional, Magnus Lie Hetland. *Edition, 2005.*

B.E. /B.Tech in Computer Science & Business Systems

Semester 3

Semester III

COMPUTATIONAL STATISTICS + Lab (continued)

Reference Books:

1. *Regression Diagnostics , Identifying Influential Data and Sources of Collinearety*, D.A. Belsey, E. Kuh and R.E. Welsch
2. *Applied Linear Regression Models*, J. Neter, W. Wasserman and M.H. Kutner.
3. *The Foundations of Factor Analysis*, A.S. Mulaik.
4. *Introduction to Linear Regression Analysis*, D.C. Montgomery and E.A. Peck.
5. *Cluster Analysis for Applications*, M.R. Anderberg.
6. *Multivariate Statistical Analysis*, D.F. Morrison.
7. *Python for Data Analysis*, Wes Mc Kinney.

B.E. /B.Tech in Computer Science & Business Systems

Semester 3

SOFTWARE ENGINEERING + Lab

Introduction: Programming in the small vs. programming in the large; software project failures and importance of software quality and timely availability; engineering approach to software development; role of software engineering towards successful execution of large software projects; emergence of software engineering as a discipline.

Software Project Management: Basic concepts of life cycle models – different models and milestones; software project planning – identification of activities and resources; concepts of feasibility study; techniques for estimation of schedule and effort; software cost estimation models and concepts of software engineering economics; techniques of software project control and reporting; introduction to measurement of software size; introduction to the concepts of risk and its mitigation; configuration management.

Software Quality and Reliability: Internal and external qualities; process and product quality; principles to achieve software quality; introduction to different software quality models like McCall, Boehm, FURPS / FURPS+, Dromey, ISO – 9126; introduction to Capability Maturity Models (CMM and CMMI); introduction to software reliability, reliability models and estimation.

Software Requirements Analysis, Design and Construction: Introduction to Software Requirements Specifications (SRS) and requirement elicitation techniques; techniques for requirement modeling – decision tables, event tables, state transition tables, Petri nets; requirements documentation through use cases; introduction to UML, introduction to software metrics and metrics based control methods; measures of code and design quality.

Object Oriented Analysis, Design and Construction: Concepts -- the principles of abstraction, modularity, specification, encapsulation and information hiding; concepts of abstract data type; Class Responsibility Collaborator (CRC) model; quality of design; design measurements; concepts of design patterns; Refactoring; object oriented construction principles; object oriented metrics.

Software Testing: Introduction to faults and failures; basic testing concepts; concepts of verification and validation; black box and white box tests; white box test coverage – code coverage, condition coverage, branch coverage; basic concepts of black-box tests – equivalence classes, boundary value tests, usage of state tables; testing use cases; transaction based testing; testing for non-functional requirements – volume, performance and efficiency; concepts of inspection.

Laboratory

B.E. /B.Tech in Computer Science & Business Systems

Semester 3

Development of requirements specification, function oriented design using SA/SD, object-oriented design using UML, test case design, implementation using C++ and testing. Use of appropriate CASE tools and other tools such as configuration management tools, program analysis tools in the software life cycle.

B.E. /B.Tech in Computer Science & Business Systems

Semester 3

Semester III

SOFTWARE ENGINEERING + Lab (continued)

Text Books:

1. *Software Engineering*, Ian Sommerville

Reference Books:

1. *Fundamentals of Software Engineering*, Carlo Ghezzi, Jazayeri Mehdi, Mandrioli Dino
2. *Software Requirements and Specification: A Lexicon of Practice, Principles and Prejudices*, Michael Jackson
3. *The Unified Development Process*, Ivar Jacobson, Grady Booch, James Rumbaugh
4. *Design Patterns: Elements of Object-Oriented Reusable Software*, Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides
5. *Software Metrics: A Rigorous and Practical Approach*, Norman E Fenton, Shari Lawrence Pfleeger
6. *Software Engineering: Theory and Practice*, Shari Lawrence Pfleeger and Joanne M. Atlee
7. *Object-Oriented Software Construction*, Bertrand Meyer
8. *Object Oriented Software Engineering: A Use Case Driven Approach* --Ivar Jacobson
9. *Touch of Class: Learning to Program Well with Objects and Contracts* --Bertrand Meyer
10. *UML Distilled: A Brief Guide to the Standard Object Modeling Language* --Martin Fowler

INDIAN CONSTITUTION (Non Credit)

(To be finalised by Respective Institute)