S. S. T. COLLEGE OF ARTS AND COMMERCE

Syllabus for F. Y. B. Sc. (Computer Science)

Program: Bachelor of Science

(Semester based Credit and Grading system)

https://www.youtube.com/sstcollege

https://www.facebook.com/sstcollegeofficial/

https://www.instagram.com/sstcollegeofficial/

http://sstcollege.edu.in/

Academic Council 24/06/2016
Item No: ________________

UNIVERSITY OF MUMBAI

Syllabus for

Program: Bachelor of Science

Course: Computer Science

With effect from

Academic Year 2016-2017

Preamble

Information and Communication Technology (ICT) has today become integral part of all industry domains as well as fields of academics and research. The industry requirements and technologies have been steadily and rapidly advancing. Organizations are increasingly opting for open source systems. The students too these days are thinking beyond career in the industry and aiming for research opportunities.

The B.Sc. Computer Science course structure therefore needed a fresh outlook and complete overhaul. A real genuine attempt has been made while designing the new syllabus for this 3-year graduate course. Not only does it prepares the students for a career in Software industry, it also motivates them towards further studies and research opportunities.

The core philosophy of overall syllabus is to -

a. Form strong foundation of Computer science,
b. Introduce emerging trends to the students in gradual way,
c. Groom the students for the challenges of ICT industry

In the first year i.e. for semester I & II, basic foundation of important skills required for software development is laid. The syllabus proposes to have four core subjects of Computer science and two core courses of Mathematics-Statistics. All core subjects are proposed to have theory as well as practical tracks. While the Computer Science courses will form fundamental skills for solving computational problems, the Mathematics & Statistics course will inculcate research-oriented acumen.

The syllabus design for further semesters encompasses more advanced and specialized courses of Computer Science.

We sincerely believe that any student taking this course will get very strong foundation and exposure to basics, advanced and emerging trends of the subject. We hope that the students' community and teachers' fraternity will appreciate the treatment given to the courses in the syllabus.

We wholeheartedly thank all experts who shared their valuable feedbacks and suggestions in order to improvise the contents, we have sincerely attempted to incorporate each of them. We further thank Chairperson and members of Board of Studies for their confidence in us.

Special thanks to Department of Computer Science and colleagues from various colleges, who volunteered or have indirectly helped designing certain specialized courses and the syllabus as a whole.

F.Y.B.Sc. Computer Science Syllabus
Credit Based System and Grading System
Academic year 2016-2017

Semester – I				
Course Code	**Course Type**	**Course Title**	**Credits**	**Lectures/Week**
USCS101	Core Subject	Computer Organization and Design	2	3
USCS102	Core Subject	Programming with Python- I	2	3
USCS103	Core Subject	Free and Open Source Software	2	3
USCS104	Core Subject	Database Systems	2	3
USCS105	Core Subject	Discrete Mathematics	2	3
USCS106	Core Subject	Descriptive Statistics and Introduction to Probability	2	3
USCS107	Ability Enhancement Course 1	Soft Skills Development	2	3
USCSP01	Core Subject Practical	Practical of USCS101 + USCS102 + USCS103+USCS104+USCS105+USCS106	6	18

Semester – II				
Course Code	**Course Type**	**Course Title**	**Credits**	**Lectures/Week**
USCS201	Core Subject	Programming with C	2	3
USCS202	Core Subject	Programming with Python– II	2	3
USCS203	Core Subject	Linux	2	3
USCS204	Core Subject	Data Structures	2	3
USCS205	Core Subject	Calculus	2	3
USCS206	Core Subject	Statistical Methods and Testing of Hypothesis	2	3
USCS207	Ability Enhancement Course 2	Green Technologies	2	3
USCSP02	Core Subject Practical	Practical of USCS201 + USCS202 + USCS203+USCS204+USCS205+USCS206	6	18

Semester I – Theory

Course: USCS101	**Computer Organization and Design (Credits : 2 Lectures/Week: 3)**	
Objectives. To understand the structure and operation of modern processors and their instruction sets **Expected Learning Outcomes:** 1) To learn about how computer systems work and underlying principles 2) To understand the basics of digital electronics needed for computers 3) To understand the basics of instruction set architecture for reduced and complex instruction sets 4) To understand the basics of processor structure and operation 5) To understand how data is transferred between the processor and I/O devices		
Unit I	**Computer Abstractions and Technology**: Basic structure and operation of a computer, functional units and their interaction. Representation of numbers and characters. **Logic circuits and functions:** Combinational circuits and functions: Basic logic gates and functions, truth tables; logic circuits and functions. Minimization with Karnaugh maps. Synthesis of logic functions with and-or-not gates, nand gates, nor gates. Fan-in and fan-out requirements; tristate buffers. Half adder, full adder, ripple carry adder. (Flip flops) Gated S-R and D latches, edge-triggered D latch. Shift registers and registers. Decoders, multiplexers. Sequential circuits and functions: State diagram and state table; finite state machines and their synthesis.	15 L
Unit II	**Instruction set architectures**: Memory organization, addressing and operations; word size, big-endian and little-endian arrangements. Instructions, sequencing. Instruction sets for RISC and CISC (examples Altera NIOS II and Freescale ColdFire). Operand addressing modes; pointers; indexing for arrays. Machine language, assembly language, assembler directives. Function calls, processor runtime stack, stack frame. Types of machine instructions: arithmetic, logic, shift, etc. Instruction sets, RISC and CISC examples.	15 L
Unit III	**Basic Processor Unit:** Main components of a processor: registers and register files, ALU, control unit, instruction fetch unit, interfaces to instruction and data memories. Datapath. Instruction fetch and execute; executing arithmetic/logic, memory access and branch instructions; hardwired and microprogrammed control for RISC and CISC. **Basic I/O:** Accessing I/O devices, data transfers between processor and I/O devices. Interrupts and exceptions: interrupt requests and processing.	15 L

Text book: 1. Carl Hamacher et al., Computer Organization and Embedded Systems, 6 ed., McGraw-Hill 2012 Additional References: 1. Patterson and Hennessy, Computer Organization and Design, Morgan Kaufmann, ARM Edition, 2011 2. R P Jain, Modern Digital Electronics, Tata McGraw Hill Education Pvt. Ltd. , 4th Edition, 2010	

Course: **USCS102**	**Programming with Python- I** **(Credits : 2 Lectures/Week: 3)**	
Objectives The objective of this paper is to introduce various concepts of programming to the students using Python. **Expected learning outcomes** 1) Students should be able to understand the concepts of programming before actually starting to write programs. 2) Students should be able to develop logic for Problem Solving. 3) Students should be made familiar about the basic constructs of programming such as data, operations, conditions, loops, functions etc. 4) Students should be able to apply the problem solving skills using syntactically simple language i.e. **Python (version: 3.X or higher)**		
Unit I	Reasons for Python as the learner's first programming language. Introduction to the IDLE interpreter (shell) and its documentation. Expression evaluation: similarities and differences compared to a calculator; expressions and operators of types int, float, boolean. Built-in function type. Operator precedence. Enumeration of simple and compound statements. The expression statement. The assert statement, whose operand is a boolean expression (values true or false). The assignment statement, dynamic binding of names to values, (type is associated with data and not with names); automatic and implicit declaration of variable names with the assignment statement; assigning the valueNone to a name. The del (delete) statement. Input/output with print and input functions. A statement list (semicolon-separated list of simple statements on a single line) as a single interpreter command. The import statement for already-defined functions and constants. The augmented assignment statement. The built-inhelp() function. Interactive and script modes of IDLE, running a script, restarting the shell. The compound statement def to define functions; the role of indentation for delimiting the body of a compound statement; calling a previously defined function. Compound data types str, tuple and list (enclosed in quotes, parentheses and brackets, respectively). Indexing individual elements within these types. Strings and tuples are immutable, lists are mutable. Built-in functions min, max, sum. Interactive solution of model problems, (e.g., finding the square root of a number or zero of a function), by repeatedly executing the body of a loop (where the body is a statement list).	15 L

Unit II	Advantages of functions, function parameters, formal parameters, actual parameters, global and local variables. The range function, the iterative for statement. The conditional statements if, if-else, if-elif-else. The iterative statements while, while-else, for-else. The continue statement to skip over one iteration of a loop, the break statement to exit the loop. Nested compound statements. Dictionaries: concept of key-value pairs, techniques to create, update and delete dictionary items. Problem-solving using compound types and statements.	15 L
Unit III	Anonymous functions. List comprehensions. Gentle introduction to object-oriented programming; using the built-in dir() function, enumerate the methods of strings, tuples, lists, dictionaries. Using these methods for problem-solving with compound types.	15 L

Text books:

1. Magnus Lie Hetland, Beginning Python: From Novice to Professional, Apress
2. Paul Gries, et al., Practical Programming: An Introduction to Computer Science Using Python 3, Pragmatic Bookshelf, 2/E 2014

Additional References:

1. Charles Dierbach, *Introduction to Computer Science using Python*, Wiley, 2013
2. Paul Gries , Jennifer Campbell, Jason Montojo, *Practical Programming: An Introduction to Computer Science Using Python 3*, Pragmatic Bookshelf, 2/E 2014
3. Adesh Pandey, *Programming Languages – Principles and Paradigms,* Narosa, 2008

Course: **USCS103**	**Free and Open-source Software** **(Credits : 2 Lectures/Week: 3)**	

Objective:

Open Source has acquired a prominent place in software industry. Having knowledge of Open Source and its related technologies is an essential for Computer Science student. This course introduces Open Source methodologies and ecosystem to students.

Expected Learning Outcome:

1) Upon completion of this course, students should have a good working knowledge of Open Source ecosystem, its use, impact and importance.
2) This course shall help student to learn Open Source methodologies, case studies with real life examples.

Unit I	**Introduction** Introduction: Open Source, Free Software, Free Software vs. Open Source software, Public Domain Software, FOSS does not mean no cost. History: BSD, The Free Software Foundation and the GNU Project. **Methodologies** Open Source History, Initiatives, Principle and methodologies. Philosophy : Software Freedom, Open Source Development Model Licenses and Patents: What Is A License, Important FOSS Licenses (Apache,BSD,GPL, LGPL), copyrights and copy lefts, Patents Economics of FOSS : Zero Marginal Cost, Income-generation opportunities, Problems with traditional commercial software, Internationalization	**15L**

	Social Impact Open source vs. closed source, Open source government, Open source ethics. Social and Financial impacts of open source technology, Shared software, Shared source, Open Source in Government.	
Unit II	**Case Studies** Example Projects: Apache web server, GNU/Linux, Android, Mozilla (Firefox), Wikipedia, Drupal, wordpress, GCC, GDB, github, Open Office. Study: Understanding the developmental models, licensings, mode of funding,commercial/non-commercial use. Open Source Hardware, Open Source Design, Open source Teaching. Open source media. **Collaboration, Community and Communication** **Contributing to Open Source Projects** Introduction to github, interacting with the community on github, Communication and etiquette, testing open source code, reporting issues, contributing code. Introduction to wikipedia, contributing to Wikipedia Or contributing to any prominent open source project of student's choice. Starting and Maintaining own Open Source Project.	**15L**
Unit III	**Understanding Open Source Ecosystem** Open Source Operating Systems: GNU/Linux, Android, Free BSD, Open Solaris. Open Source Hardware, Virtualization Technologies, Containerization Technologies: Docker, Development tools, IDEs, debuggers, Programming languages, LAMP, Open Source database technologies	**15L**

Text books:

1. Unix Concepts and Applications by Sumitabha Das, Tata McGraw Hill Education, 2006
2. The official Ubuntu Book, 8th Edition

Additional references:

1. The Linux Documentation Project: http://www.tldp.org/
2. Docker Project Home: http://www.docker.com
3. Linux kernel Home: http://kernel.org
4. Open Source Initiative: https://opensource.org/
5. Linux Documentation Project: http://www.tldp.org/
6. Wikipedia: https://en.wikipedia.org/
7. https://en.wikipedia.org/wiki/Wikipedia:Contributing_to_Wikipedia
8. Github: https://help.github.com/
9. The Linux Foundation: http://www.linuxfoundation.org/

Course: **USCS104**	**Database Systems** **(Credits : 2 Lectures/Week: 3)**	
Objectives: The objective of this course is to introduce the concept of the DBMS with respect to the relational model, to specify the functional and data requirements for a typical database application and to understand creation, manipulation and querying of data in databases **Expected Learning Outcomes** 1) Students should be able to evaluate business information problem and find the requirements of a problem in terms of data. 2) Students should be able to design the database schema with the use of appropriate data types for storage of data in database. 3) Students should be able to create, manipulate, query and back up the databases.		
Unit I	**Introduction to DBMS –** Database, DBMS – Definition, Overview of DBMS, Advantages of DBMS, Levels of abstraction, Data independence, DBMS Architecture **Data models** - Client/Server Architecture, Object Based Logical Model, Record Based Logical Model (relational, hierarchical, network) **Entity Relationship Model -** Entities, attributes, entity sets, relations, relationship sets, Additional constraints (key constraints, participation constraints, weak entities, aggregation / generalization, Conceptual Design using ER (entities VS attributes, Entity Vs relationship, binary Vs ternary, constraints beyond ER) **Relational data model**– Domains, attributes, Tuples and Relations, Relational Model Notation, Characteristics of Relations, Relational Constraints - primary key, referential integrity, unique constraint, Null constraint, Check constraint **ER to Table**- Entity to Table, Relationship to tables with and without key constraints.	15L
Unit II	**Schema refinement and Normal forms:** Functional dependencies, first, second, third, and BCNF normal forms based on primary keys, lossless join decomposition. **Relational Algebra** operations (selection, projection, set operations union, intersection, difference, cross product, Joins –conditional, equi join and natural joins, division) **DDL Statements** - Creating Databases, Using Databases, datatypes, Creating Tables (with integrity constraints – primary key, default, check, not null), Altering Tables, Renaming Tables, Dropping Tables, Truncating Tables, Backing Up and Restoring databases **DML Statements** – Viewing the structure of a table insert, update, delete, Select all columns, specific columns, unique records, conditional select, in clause, between clause, limit, aggregate functions (count, min, max, avg, sum), group by clause, having clause	15L

Unit III	**Functions** – String Functions (concat, instr, left, right, mid, length, lcase/lower, ucase/upper, replace, strcmp, trim, ltrim, rtrim), Math Functions (abs, ceil, floor, mod, pow, sqrt, round, truncate) Date Functions (adddate, datediff, day, month, year, hour, min, sec, now, reverse) **Joining Tables** – inner join, outer join (left outer, right outer, full outer) **Subqueries** – subqueries with IN, EXISTS, subqueries restrictions, Nested subqueries, ANY/ALL clause, correlated subqueries **Database Protection:** Security Issues, Threats to Databases, Security Mechanisms, Role of DBA, Discretionary Access Control **Views** (creating, altering dropping, renaming and manipulating views) **DCL Statements** (creating/dropping users, privileges introduction, granting/revoking privileges, viewing privileges)	15L

Text books:

1. Ramez Elmasri & Shamkant B.Navathe, Fundamentals of Database Systems, Pearson Education, Sixth Edition, 2010
2. Ramakrishnam, Gehrke, Database Management Systems, McGraw-Hill, 2007
3. Joel Murach, Murach's MySQL, Murach, 2012

Additional References:

1. Robert Sheldon, Geoff Moes, Begning MySQL, Wrox Press, 2005.

Course: **USCS105**	**Discrete Mathematics** **(Credits : 2 Lectures/Week: 3)**	

Objectives:

The purpose of the course is to familiarize the prospective learners with mathematical structures that are fundamentally discrete. This course introduces sets and functions, forming and solving recurrence relations and different counting principles. These concepts are useful to study or describe objects or problems in computer algorithms and programming languages.

Expected Learning Outcomes:

1) To provide overview of theory of discrete objects, starting with relations and partially ordered sets.
2) Study about recurrence relations, generating function and operations on them.
3) Give an understanding of graphs and trees, which are widely used in software.
4) Provide basic knowledge about models of automata theory and the corresponding formal languages.

Unit I	**Recurrence Relations** **(a) Functions:** Definition of function. Domain, co domain and the range of a function. Direct and inverse images. Injective, surjective and bijective functions. Composite and inverse functions. **(b) Relations**: Definition and examples. Properties of relations , Partial Ordering sets, Linear Ordering Hasse Daigrams , Maximum and Minimum elements, Lattices **(c) Recurrence Relations:** Definition of recurrence relations, Formulating recurrence relations, solving recurrence relations- Back tracking method, Linear homogeneous recurrence relations with constant coefficients. Solving linear homogeneous recurrence relations with constant coefficients of degree two when characteristic equation has distinct roots and only one root, Particular solutions of non linear homogeneous recurrence relation, Solution of recurrence relation by the method of generation functions, Applications- Formulate and solve recurrence relation for Fibonacci numbers, Tower of Hanoi, Intersection of lines in a plane, Sorting Algorithms.	15L
Unit II	**Counting Principles , Languages and Finite State Machine** **(a) Permutations and Combinations:** Partition and Distribution of objects, Permutation with distinct and indistinct objects, Binomial numbers, Combination with identities: Pascal Identity, Vandermonde's Identity, Pascal triangle, Binomial theorem, Combination with indistinct objects. **(b) Counting Principles:** Sum and Product Rules, Two-way counting, Tree diagram for solving counting problems, Pigeonhole Principle (without proof); Simple examples, Inclusion Exclusion Principle (Sieve formula) (Without proof). **(c) Languages, Grammars and Machines:** Languages , regular Expression and Regular languages, Finite state Automata, grammars, Finite state machines, Gödel numbers, Turing machines.	15L
Unit III	**Graphs and Trees** **(a) Graphs :** Definition and elementary results, Adjacency matrix, path matrix, Representing relations using diagraphs, Warshall's algorithm- shortest path , Linked representation of a graph, Operations on graph with algorithms - searching in a graph; Insertion in a graph, Deleting from a graph, Traversing a graph- Breadth-First search and Depth-First search. **(b) Trees:** Definition and elementary results. Ordered rooted tree, Binary trees, Complete and extended binary trees, representing binary trees in memory, traversing binary trees, binary search tree, Algorithms for searching and inserting in binary search trees, Algorithms for deleting in a binary search tree	15L

Textbook: 1. Discrete Mathematics and Its Applications, Seventh Edition by Kenneth H. Rosen, McGraw Hill Education (India) Private Limited. (2011) 2. Norman L. Biggs, Discrete Mathematics, Revised Edition, Clarendon Press, Oxford 1989. 3. Data Structures Seymour Lipschutz, Schaum's out lines, McGraw- Hill Inc. **Additional References**: 1. Elements of Discrete Mathematics: C.L. Liu , Tata McGraw- Hill Edition . 2. Concrete Mathematics (Foundation for Computer Science): Graham, Knuth, Patashnik Second Edition, Pearson Education. 3. Discrete Mathematics: Semyour Lipschutz, Marc Lipson, Schaum's out lines, McGraw- Hill Inc. 4. Foundations in Discrete Mathematics: K.D. Joshi, New Age Publication, New Delhi.

Course: **USCS106**	**Descriptive Statistics and Introduction to Probability (Credits : 2 Lectures/Week: 3)**	
Objectives: The purpose of this course is to familiarize students with basics of Statistics. This will be essential for prospective researchers and professionals to know these basics. **Expected Learning Outcomes**: 1) Enable learners to know descriptive statistical concepts 2) Enable study of probability concept required for Computer learners		
Unit I	**Data Presentation** Data types : attribute, variable, discrete and continuous variable Data presentation : frequency distribution, histogram o give, curves, stem and leaf display **Data Aggregation** Measures of Central tendency: Mean, Median, mode for raw data, discrete, grouped frequency distribution. Measures dispersion: Variance, standard deviation, coefficient of variation for raw data, discrete and grouped frequency distribution, quartiles, quantiles Real life examples	15L
Unit II	**Moments:** raw moments, central moments, relation between raw and central moments **Measures of Skewness and Kurtosis:** based on moments, quartiles, relation between mean, median, mode for symmetric, asymmetric frequency curve. **Correlation and Regression:** bivariate data, scatter plot, correlation, nonsense correlation, Karl pearson's coefficients of correlation, independence. **Linear regression:** fitting of linear regression using least square regression, coefficient of determination, properties of regression coefficients (only statement)	15L

Unit III	**Probability :** Random experiment, sample space, events types and operations of events **Probability definition :** classical, axiomatic, Elementary Theorems of probability (without proof) – $0 \le P(A) \le 1$, – $P(A \cup B) = P(A) + P(B) - P(A \cap B)$ – $P(A') = 1 - P(A)$ – $P(A) \le P(B)$ if $A \subset B$ Conditional probability, 'Bayes' theorem, independence, Examples on Probability	15L

Text Book:

1. Trivedi, K.S.(2001) : Probability, Statistics, Design of Experiments and Queuing theory, with applications of Computer Science, Prentice Hall of India, New Delhi

Additional References:

1. Ross, S.M. (2006): A First course in probability. 6th Edn Pearson
2. Kulkarni, M.B., Ghatpande, S.B. and Gore, S.D. (1999): common statistical tests. Satyajeet Prakashan, Pune
3. Gupta, S.C. and Kapoor, V.K. (1987): Fundamentals of Mathematical Statistics, S. Chand and Sons, New Delhi
4. Gupta, S.C. and Kapoor, V.K. (1999): Applied Statistics, S. Chand and Son's, New Delhi
5. Montgomery, D.C. (2001): Planning and Analysis of Experiments, wiley.

Course: USCS107	**Soft Skills Development (Credits : 2 Lectures/Week: 3)**	

Objectives:

To help learners develop their soft skills and develop their personality together with their technical skills. Developing professional, social and academic skills to harness hidden strengths, capabilities and knowledge equip them to excel in real work environment and corporate life. Understand various issues in personal and profession communication and learn to overcome them

Expected Learning Outcomes:

1) To know about various aspects of soft skills and learn ways to develop personality
2) Understand the importance and type of communication in personal and professional environment.
3) To provide insight into much needed technical and non-technical qualities in career planning.
4) Learn about Leadership, team building, decision making and stress management

Unit I	**Introduction to Soft Skills and Hard Skills** **Personality Development:** Knowing Yourself, Positive Thinking, Johari's Window, Communication Skills, Non-verbal Communication, Physical Fitness **Emotional Intelligence:** Meaning and Definition, Need for Emotional Intelligence, Intelligence Quotient versus Emotional Intelligence Quotient, Components of Emotional Intelligence, Competencies of Emotional Intelligence, Skills to Develop Emotional Intelligence **Etiquette and Mannerism:** Introduction, Professional Etiquette, Technology Etiquette **Communication Today:** Significance of Communication, GSC's 3M Model of Communication, Vitality of the Communication Process, Virtues of Listening, Fundamentals of Good Listening, Nature of Non-Verbal Communication, Need for Intercultural Communication, Communicating Digital World	15L
Unit II	**Academic Skills** **Employment Communication:** Introduction, Resume, Curriculum Vitae, Scannable Resume, Developing an Impressive Resume, Formats of Resume, Job Application or Cover Letter **Professional Presentation:** Nature of Oral Presentation, Planning a Presentation, Preparing the Presentation, Delivering the Presentation **Job Interviews:** Introduction, Importance of Resume, Definition of Interview, Background Information, Types of Interviews, Preparatory Steps for Job Interviews, Interview Skill Tips, Changes in the Interview Process, FAQ During Interviews **Group Discussion:** Introduction, Ambience/Seating Arrangement for Group Discussion, Importance of Group Discussions, Difference between Group Discussion, Panel Discussion and Debate, Traits, Types of Group Discussions, topic based and Case based Group Discussion, Individual Traits	15L
Unit III	**Professional Skills** **Creativity at Workplace:** Introduction, Current Workplaces, Creativity, Motivation, Nurturing Hobbies at Work, The Six Thinking Hat Method **Ethical Values:** Ethics and Society, Theories of Ethics, Correlation between Values and Behavior, Nurturing Ethics, Importance of Work Ethics, Problems in the Absence of Work Ethics **Capacity Building: Learn, Unlearn and Relearn**: Capacity Building, Elements of Capacity Building, Zones of Learning, Ideas for Learning, Strategies for Capacity Building **Leadership and Team Building:** Leader and Leadership, Leadership Traits, Culture and Leadership, Leadership Styles and Trends, Team Building, Types of Teams, **Decision Making and Negotiation:** Introduction to Decision Making, Steps for Decision Making, Decision Making Techniques, Negotiation Fundamentals, Negotiation Styles, Major Negotiation Concepts **Stress and Time Management:** Stress, Sources of Stress, Ways to Cope with Stress	15L

Text book: 1. *Soft Skills: an Integrated Approach to Maximise Personality,* Gajendra S. Chauhan, Sangeeta Sharma, Wiley India **Additional References:** 1. *Personality Development and Soft Skills*, Barun K. Mitra, Oxford Press 2. *Business Communication,* Shalini Kalia, Shailja Agrawal, Wiley India 3. *Soft Skills - Enhancing Employability*, M. S. Rao, I. K. International 4. *Cornerstone: Developing Soft Skills*, Sherfield, Pearson India

Semester I – Practical

Course: **USCSP1**	**Practical of USCS101 + USCS102 + USCS103+USCS104+USCS105+USCS106 (Credits : 6, Lectures/Week: 18)**	
USCSP101	**Computer Organization and Design** 1. Study and verify the truth table of various logic gates (NOT, AND, OR, NAND, NOR, EX-OR, and EX-NOR). 2. Simplify given Boolean expression and realize it. 3. Design and verify a half/full adder 4. Design and verify half/full subtractor 5. Design a 4 bit magnitude comparator using combinational circuits. 6. Design and verify the operation of flip-flops using logic gates. 7. Verify the operation of a counter. 8. Verify the operation of a 4 bit shift register 9. Using SPIM, write and test an adding machine program that repeatedly reads in integers and adds them into a running sum. The program should stop when it gets an input that is 0, printing out the sum at that point. 10. Using SPIM, write and test a program that reads in a positive integer using the SPIM system calls. If the integer is not positive, the program should terminate with the message "Invalid Entry"; otherwise the program should print out the names of the digits of the integers, delimited by exactly one space. For example, if the user entered "528," the output would be "Five Two Eight." *# Practical No. 1 to 8 can be performed using any open source simulator (like Logisim)* *(Download it from https://sourceforge.net/projects/circuit/)* # Practical No. 9 and 10 are required to be done using SPIM. SPIM is a self-contained simulator that will run MIPS R2000/R3000 assembly language programs. # Latest version is available at https://sourceforge.net/projects/spimsimulator/	

USCSP102	**Programming with Python – I** 1. Installing and setting up the Python IDLE interpreter. Executing simple statements like expression statement (numeric and Boolean types), assert, assignment, delete statements; the print function for output. 2. Script and interactive modes; defining a function in the two modes; executing a script; interactively executing a statement list (semicolon-separated sequence of simple statements); the input function. 3. Programs based on lists, conditional constructs, the for statement and the range function; interactively using the built-in functions len, sum, max, min 4. Programs related to string manipulation 5. Programs based on the while statement; importing and executing built-in functions from the time, math and random modules 6. Programs using break and continue statements. 7. Programs related to dictionaries 8. Programs using list comprehenstions and anonymous functions 9. Programs using the built-in methods of the string, list and dictionary classes	

USCSP103	**Free and Open Source Software** 1. Identify any Open Source software and create detailed report about it. Sample Guidelines. a. Idea b. What problem does it solves? c. Licensing model d. Intent behind making it open source e. Monetization models f. Popularity g. Impact 2. Learn at least three different open source licenses and create a brief report about them. a. History of license b. Idea c. What problems does it solve? d. Detailed licensing model e. Which popular software are released under this license? f. Any popular news associated with this license? g. Popularity h. Impact 3. Contributing to Open Source a. Identify any Open Source project of your interest b. Learn more about the project w.r.t. Lab 1. c. Start contributing to the project either by i. Testing ii. Reporting bugs iii. Coding iv. Helping in documentation v. Participating in discussions vi. Participating in pre-release testing programs vii. UI development. viii. Or any other important area. 4. Hands on with Open Source Software a. Identify any open source software of your interest b. Learn it from practical view-point c. Give a brief presentation about it to the class d. Sample projects: gcc, gdb, drupal, wordpress, apache web server, mysql database 5. Contributing to Wikipedia: a. Introduction to wikipedia: operating model, license, how to contribute? b. Create your user account on wikipedia c. Identify any topic of your choice and contribute the missing information 6. Github a. Create and publish your own open source project: Write any simple program using your choice of programming language.	

USCSP103	b. Create a repository on github and save versions of your project. You'll learn about the staging area, committing your code, branching, and merging, c. Using GitHub to Collaborate: Get practice using GitHub or other remote repositories to share your changes with others and collaborate on multi-developer projects. You'll learn how to make and review a pull request on GitHub. d. Contribute to a Live Project: Students will publish a repository containing their reflections from the course and submit a pull request. 7. Open Source Operating Systems a. Learn any open source operating system of your choice : Linux, Android, FreeBSD, Open Solaris etc. b. Learn the installation. c. Identify the unique features of the OS of your choice. 8. Virtualization: Open Source virtualization technologies: a. Install and configure any one: VirtualBox, Zen, KVM b. Create and use virtual machines 9. Containerization: a. Containerization technologies: docker, rocket, LXD b. Install and configure any containerization technology c. Create and use containers using it 10. Linux Kernel: Learn Linux kernel with respect to: a. What is Linux kernel? b. Operating model c. Licensing Model d. How development works? e. Download kernel source code. f. Compile the Kernel	
USCSP104	**Database Systems** 1. For given scenario • Draw E-R diagram and convert entities and relationships to table. 2. Write relational algebra queries on the tables created in Practical-1. 3. Perform the following: • Viewing all databases • Creating a Database • Viewing all Tables in a Database • Creating Tables (With and Without Constraints) • Inserting/Updating/Deleting Records in a Table • Saving (Commit) and Undoing (rollback) 4. Perform the following: • Altering a Table • Dropping/Truncating/Renaming Tables • Backing up / Restoring a Database	

	5. Perform the following: • Simple Queries • Simple Queries with Aggregate functions • Queries with Aggregate functions (group by and having clause) 6. Queries involving • Date Functions • String Functions • Math Functions 7. Join Queries • Inner Join • Outer Join 8. Subqueries • With IN clause • With EXISTS clause 9. Views • Creating Views (with and without check option) • Dropping views • Selecting from a view 10. DCL statements • Granting and revoking permissions	
USCSP105	**Discrete Mathematics** 1. Graphs of standard functions such as absolute value function, inverse function, logarithmic and exponential functions, flooring and ceiling functions, trigonometric functions over suitable intervals. 2. Partial ordering sets, Hasse diagram and Lattices. 3. Recurrence relation. 4. Different counting principles. 5. Finite state Automata and Finite state machines. 6. Warshall's Algorithm. 7. Shortest Path algorithms. 8. Operations on graph. 9. Breadth and Depth First search algorithms. 10. Concept of searching, inserting and deleting from binary search trees.	
USCSP106	**Descriptive Statistics and Introduction to Probability** *(To be implemented using R)* 1. Frequency distribution and data presentation 2. Measures of central tendency 3. Data entry using, functions, c(), scan (), Creating vectors, Mathematical Operations: ** +/-/*/ / ^ , exp, log, log10, etc, creating vector of text type, useful functions: data, frame, matrix operations, seq(), split() etc. 4. Frequency distribution using cut(), table() 5. Data presentation 6. Summary Statistics (measures of central tendency, dispersion) 7. Measures of skewness and kurtosis 8. Correlation and regression 9. Probability 10. Conditional probability	

Semester II - Theory

Course: **USCS201**	**Programming with C** **(Credits : 2 Lectures/Week: 3)**	
Objectives: The objective of this course is to provide a comprehensive study of the C programming language, stressing upon the strengths of C, which provide the students with the means of writing modular, efficient, maintainable, and portable code. **Expected Learning Outcomes** 1) Students should be able to write, compile and debug programs in C language. 2) Students should be able to use different data types in a computer program. 3) Students should be able to design programs involving decision structures, loops and functions. 4) Students should be able to explain the difference between call by value and call by reference 5) Students should be able to understand the dynamics of memory by the use of pointers. 6) Students should be able to use different data structures and create/update basic data files.		
Unit I	**Structure of C program**: Header and body, Use of comments. Interpreters vs compilers, Python vs C. Compilation of a program. Formatted I/O: printf(), scanf(). **Data**: Variables, Constants, data types like: int, float char, double and void, short and long size qualifiers, signed and unsigned qualifiers. Compare with datatypes in Python. Compare static typing in C vs dynamic typing in Python **Variables**: Declaring variables, scope of the variables according to block, hierarchy of data types. Compare explicit declarations in C with implicit declarations in Python. **Types of operators**: Arithmetic, relational, logical, compound assignment, increment and decrement, conditional or ternary, bitwise and comma operators. Precedence and order of evaluation, statements and Expressions. Automatic and explicit type conversion. **Iterations**: Control statements for decision making: (i) Branching: if statement, else.. if statement, (does the writer mean if-else or nested ifs)switch statement. (ii) Looping: while loop, do.. while, for loop. (iii) Jump statements: break, continue and goto.	15L
Unit II	**Arrays:** (One and two dimensional), declaring array variables, initialization of arrays, accessing array elements. Compare array types of C with list and tuple types of Python. **Data Input and Output functions**: Character I/O format: getch(), getche(), getchar(), getc(), gets(), putchar(), putc(), puts(). **Manipulating Strings**: Declaring and initializing String variables, Character and string handling functions. Compare with Python strings. **Functions**: Function declaration, function definition, Global and local variables, return statement, Calling a function by passing values. **Recursion**: Definition, Recursive functions.	15L

Unit III	**Pointer:** Fundamentals, Pointer variables, Referencing and de-referencing, Pointer Arithmetic, Using Pointers with Arrays, Using Pointers with Strings, Array of Pointers, Pointers as function arguments, Functions returning pointers. **Dynamic Memory Allocation**: malloc(), calloc(), realloc(), free() and sizeof operator. Compare with automatic garbage collection in Python. **Structure**: Declaration of structure, reading and assignment of structure variables, Array of structures, arrays within structures, structures within structures. Compare C structures with Python tuples. **Unions**: Defining and working with unions. **File handling**: Different types of files like text and binary, Different types of functions: fopen(), fclose(), fgetc(), fputc(), fgets(), fputs(), fscanf(), fprintf(), getw(), putw(), fread(), fwrite(), fseek().	15L

Text books:

1. Programming in ANSI C (Third Edition) : E Balagurusamy, TMH

Additional References:

1. Pradip Dey, Manas Ghosh, “Programming in C”, second edition, Oxford University Press
2. Yashavant P. Kanetkar. “ Let Us C”, BPB Publications

Course: **USCS202**	**Programming with Python – II** **(Credits : 2 Lectures/Week: 3)**	

Objective:
The objective of this paper is to explore the style of structured programming to give the idea to the students how programming can be used for designing real-life applications by reading/writing to files, GUI programming, interfacing database/networks and various other features.

Expected Learning Outcomes

1) Students should be able to understand how to read/write to files using python.
2) Students should be able to catch their own errors that happen during execution of programs.
3) Students should get an introduction to the concept of pattern matching.
4) Students should be made familiar with the concepts of GUI controls and designing GUI applications.
5) Students should be able to connect to the database to move the data to/from the application.
6) 6)Students should know how to connect to computers, read from URL and send email.

Unit I	**Python File Input-Output:** Opening and closing files, various types of file modes, reading and writing to files, manipulating directories. Iterables, iterators and their problemsolving applications. **Exception handling:** What is an exception, various keywords to handle exceptions such try, catch, except, else, finally, raise. **Regular Expressions:** Concept of regular expression, various types of regular expressions, using match function.	15 L
Unit II	**GUI Programming in Python (using Tkinter/wxPython/Qt)** What is GUI, Advantages of GUI, Introduction to GUI library. Layout management, events and bindings, fonts, colours, drawing on canvas (line, oval, rectangle, etc.) Widgets such as : frame, label, button, checkbutton, entry, listbox, message, radiobutton, text, spinbox etc	15 L
Unit III	**Database connectivity in Python**: Installing mysql connector, accessing connector module module, using connect, cursor, execute & close functions, reading single & multiple results of query execution, executing different types of statements, executing transactions, understanding exceptions in database connectivity. **Network connectivity**: Socket module, creating server-client programs, sending email, reading from URL	15 L

Text books:

1. Paul Gries , Jennifer Campbell, Jason Montojo, *Practical Programming: An Introduction to Computer Science Using Python 3*, Pragmatic Bookshelf, 2/E 2014

Additional References:

1. James Payne , *Beginning Python: Using Python 2.6 and Python 3,* Wiley India, 2010
2. A. Lukaszewski, MySQL for Python: Database Access Made Easy, Pact Publisher, 2010

Course: **USCS203**	**Linux** **(Credits : 2 Lectures/Week: 3)**	
Objectives: This course introduces various tools and techniques commonly used by Linux programmers, system administrators and end users to achieve their day to day work in Linux environment. It is designed for computer students who have limited or no previous exposure to Linux. **Expected Learning Outcomes:** 1) Upon completion of this course, students should have a good working knowledge of Linux, from both a graphical and command line perspective, allowing them to easily use any Linux distribution. 2) This course shall help student to learn advanced subjects in computer science practically. 3) Student shall be able to progress as a Developer or Linux System Administrator using the acquired skill set.		
Unit I	**Introduction** History of Linux, Philosophy, Community, Terminology, Distributions, Linux kernel vs distribution. Why learn Linux? Importance of Linux in software ecosystem: web servers, supercomputers, mobile, servers. **Installation** Installation methods, Hands on Installation using CD/DVD or USB drive. **Linux Structure** Linux Architecture, Filesystem basics, The boot process, init scripts, runlevels, shutdown process, Very basic introductions to Linux processes, Packaging methods: rpm/deb, Graphical Vs Command line.	15L
Unit II	**Graphical Desktop** Session Management, Basic Desktop Operations, Network Management, Installing and Updating Software, Text editors: gedit, vi, vim, emacs, Graphics editors, Multimedia applications. **Command Line** Command line mode options, Shells, Basic Commands, General Purpose Utilities, Installing Software, User management, Environment variables, Command aliases. **Linux Documentation** man pages, GNU info, help command, More documentation sources **File Operations** Filesystem, Filesystem architecture, File types, File attributes, Working with files, Backup, compression	15L
Unit III	**Security** Understanding Linux Security, Uses of root, sudo command, working with passwords, Bypassing user authentication, Understanding ssh **Networking** Basic introduction to Networking, Network protocols: http, ftp etc., IP address, DNS, Browsers, Transferring files. ssh, telnet, ping, traceroute, route, hostname, networking GUI. **Basic Shell Scripting** Features and capabilities, Syntax, Constructs, Modifying files, Sed, awk command, File manipulation utilities, Dealing with large files and Text, String manipulation, Boolean expressions, File tests, Case, Debugging, Regular expressions	15L

Text book:

1) Unix Concepts and Applications by Sumitabha Das.
2) Official Ubuntu Book, 8th Edition, by Matthew Helmke & Elizabeth K. Joseph with Jose Antonio Rey and Philips Ballew, Prentice Hall

Additional References:

1) Linux kernel Home: http://kernel.org
2) Open Source Initiative: https://opensource.org/
3) The Linux Foundation: http://www.linuxfoundation.org/

Course: USCS204	**Data Structures (Credits : 2 Lectures/Week: 3)**	
Objectives: To explore and understand the concepts of Data Structures and its significance in programming. Provide and holistic approach to design, use and implement abstract data types. Understand the commonly used data structures and various forms of its implementation for different applications using Python. **Expected Learning Outcomes:** 1) Learn about Data structures, its types and significance in computing 2) Explore about Abstract Data types and its implementation 3) Ability to program various applications using different data structure in Python		
Unit I	**Abstract Data Types:** Introduction, The Date Abstract Data Type, Bags, Iterators. Application **Arrays:** Array Structure, Python List, Two Dimensional Arrays, Matrix Abstract Data Type, Application **Sets and Maps:** Sets-Set ADT, Selecting Data Structure, List based Implementation, Maps-Map ADT, List Based Implementation, Multi-Dimensional Arrays-Multi-Array ADT, Implementing Multiarrays, Application **Algorithm Analysis:** Complexity Analysis-Big-O Notation, Evaluating Python Code, Evaluating Python List, Amortized Cost, Evaluating Set ADT, Application **Searching and Sorting:** Searching-Linear Search, Binary Search, Sorting-Bubble, Selection and Insertion Sort, Working with Sorted Lists-Maintaining Sorted List, Maintaining sorted Lists.	15L
Unit II	**Linked Structures**: Introduction, Singly Linked List-Traversing, Searching, Prepending and Removing Nodes, Bag ADT-Linked List Implementation. Comparing Implementations, Linked List Iterators, More Ways to Build Kinked Lists, Applications-Polynomials **Stacks**: Stack ADT, Implementing Stacks-Using Python List, Using Linked List, Stack Applications-Balanced Delimiters, Evaluating Postfix Expressions **Queues**: Queue ADT, Implementing Queue-Using Python List, Circular Array, Using List, Priority Queues- Priority Queue ADT, Bounded and unbounded Priority Queues **Advanced Linked List**: Doubly Linked Lists-Organization and Operation, Circular Linked List-Organization and Operation, Multi Lists	15L

Unit III	**Recursion:** Recursive Functions, Properties of Recursion, Its working, Recursive Applications **Hash Table:** Introduction, Hashing-Linear Probing, Clustering, Rehashing, Separate Chaining, Hash Functions **Advanced Sorting:** Merge Sort, Quick Sort, Radix Sort, Sorting Linked List **Binary Trees:** Tree Structure, Binary Tree-Properties, Implementation and Traversals, Expression Trees, Heaps and Heapsort,Search Trees	15L

Text book:

1) *Data Structure and algorithm Using Python*, Rance D. Necaise, 2016 Wiley India Edition
2) *Data Structure and Algorithm in Python*, Michael T. Goodrich, Robertom Tamassia, M. H. Goldwasser, 2016 Wiley India Edition

Additional References:

1) *Data Structure and Algorithmic Thinking with Python*- Narasimha Karumanchi, 2015, Careermonk Publications
2) Fundamentals of Python: Data Structures, Kenneth Lambert, Delmar Cengage Learning

Course: **USCS205**	**Calculus** **(Credits : 2 Lectures/Week: 3)**	

Objectives:

The course is designed to have a grasp of important concepts of Calculus in a scientific way. It covers topics from as basic as definition of functions to partial derivatives of functions in a gradual and logical way. The learner is expected to solve as many examples as possible to a get compete clarity and understanding of the topics covered.

Expected Learning Outcomes:

1) Understanding of Mathematical concepts like limit, continuity, derivative, integration of functions.
2) Ability to appreciate real world applications which uses these concepts.
3) Skill to formulate a problem through Mathematical modeling and simulation.

Unit I	**DERIVATIVES AND ITS APPLICATIONS:** Review of Functions, limit of a function, continuity of a function, derivative function. Derivative In Graphing And Applications: Analysis of Functions: Increase, Decrease, Concavity, Relative Extrema; Graphing Polynomials, Rational Functions, Cusps and Vertical Tangents. Absolute Maxima and Minima, Applied Maximum and Minimum Problems, Newton's Method.	15L
Unit II	**INTEGRATION AND ITS APPLICATIONS:** An Overview of the Area Problem, Indefinite Integral, Definition of Area as a Limit; Sigma Notation, Definite Integral, Evaluating Definite Integrals by Substitution, Area Between Two Curves, Length of a Plane Curve. Numerical Integration: Simpson's Rule. Modeling with Differential Equations, Separation of Variables, Slope Fields, Euler's Method, First-Order Differential Equations and Applications.	15L
Unit III	**PARTIAL DERIVATIVES AND ITS APPLICATIONS:** Functions of Two or More Variables Limits and Continuity Partial Derivatives, Differentiability, Differentials, and Local Linearity, Chain Rule, Directional Derivatives and Gradients, Tangent Planes and Normal, Vectors, Maxima and Minima of Functions of Two Variables.	15L

Textbook:

1. Calculus: Early transcendental (10th Edition): Howard Anton, Irl Bivens, Stephen Davis, John Wiley & sons, 2012.

Additional References:

1. Calculus and analytic geometry (9th edition): George B Thomas, Ross L Finney, Addison Wesley, 1995
2. Calculus: Early Transcendentals (8th Edition): James Stewart, Brooks Cole, 2015.
3. Calculus (10th Edition): Ron Larson, Bruce H. Edwards, Cengage Learning, 2013.
4. Thomas' Calculus (13th Edition): George B. Thomas, Maurice D. Weir, Joel R. Hass, Pearson, 2014.

Course: **USCS206**	**Statistical Methods and Testing of Hypothesis (Credits : 2 Lectures/Week: 3)**	

Objectives:
The purpose of this course is to familiarize students with basics of Statistics. This will be essential for prospective researchers and professionals to know these basics.

Expected Learning Outcomes:

1) Enable learners to know descriptive statistical concepts
2) Enable study of probability concept required for Computer learners

Unit I	**Standard distributions:** random variable; discrete, continuous, expectation and variance of a random variable, pmf, pdf, cdf, reliability, Introduction and properties without proof for following distributions; binomial, normal, chi-square, t, F. Examples	15L
Unit II	**Hypothesis testing:** one sided, two sided hypothesis, critical region, p-value, tests based on t, Normal and F, confidence intervals. Analysis of variance : one-way, two-way analysis of variance	15L
Unit III	**Non-parametric tests:** need of non-parametric tests, sign test, Wilicoxon's signed rank test, run test, Kruskal-Walis tests. Post-hoc analysis of one-way analysis of variance : Duncan's test Chi-square test of association	15L

Text Book:

1. Trivedi, K.S.(2009) : Probability, Statistics, Design of Experiments and Queuing theory, with applications of Computer Science, Prentice Hall of India, New Delhi

Additional References:

1. Ross, S.M. (2006): A First course in probability. 6th Ed^n Pearson
2. Kulkarni, M.B., Ghatpande, S.B. and Gore, S.D. (1999): Common statistical tests.
   Satyajeet Prakashan, Pune
3. Gupta, S.C. and Kapoor, V.K. (2002) : Fundamentals of Mathematical Statistics,
   S. Chand and Sons, New Delhi
4. Gupta, S.C. and Kapoor, V.K. (4th Edition) : Applied Statistics, S. Chand and Son's, New Delhi
5. Montgomery, D.C. (2001): Planning and Analysis of Experiments, Wiley.

Course: USCS207	**Green Technologies (Credits : 2 Lectures/Week: 3)**	

Objectives:
To familiarize with the concept of Green Computing and Green IT infrastructure for making computing and information system environment sustainable. Encouraging optimized software and hardware designs for development of Green IT Storage, Communication and Services. To highlight useful approaches to embrace green IT initiatives.

Expected Learning Outcomes:

1) Learn about green IT can be achieved in and by hardware, software, network communication and data center operations.
2) Understand the strategies, frameworks, processes and management of green IT

Unit I	**Green IT Overview:** Introduction , Environmental Concerns and Sustainable Development, Environmental Impacts of IT, Green I , Holistic Approach to Greening IT, Greening IT, Applying IT for Enhancing Environmental Sustainability, Green IT Standards and Eco-Labelling of IT , Enterprise Green IT Strategy, Green Washing, Green IT: Burden or Opportunity? **Green Devices and Hardware:** Introduction , Life Cycle of a Device or Hardware, Reuse, Recycle and Dispose **Green Software:** Introduction , Processor Power States , Energy-Saving Software Techniques, Evaluating and Measuring Software Impact to Platform Power **Sustainable Software Development:** Introduction, Current Practices, Sustainable Software, Software Sustainability Attributes, Software Sustainability Metrics, Sustainable Software Methodology, Defining Actions	15L
Unit II	**Green Data Centres:** Data Centres and Associated Energy Challenges, Data Centre IT Infrastructure, Data Centre Facility Infrastructure: Implications for Energy Efficiency, IT Infrastructure Management, Green Data Centre Metrics **Green Data Storage:** Introduction , Storage Media Power Characteristics, Energy Management Techniques for Hard Disks, System-Level Energy Management **Green Networks and Communications:** Introduction, Objectives of Green Network Protocols, Green Network Protocols and Standards **Enterprise Green IT Strategy:** Introduction, Approaching Green IT Strategies, Business Drivers of Green IT Strategy, Business Dimensions for Green IT Transformation, Organizational Considerations in a Green IT Strategy, Steps in Developing a Green IT Strategy, Metrics and Measurements in Green Strategies.	15L
Unit III	**Sustainable Information Systems and Green Metrics:** Introduction, Multilevel Sustainable Information, Sustainability Hierarchy Models, Product Level Information, Individual Level Information, Functional Level Information, Organizational Level Information, Measuring the Maturity of Sustainable ICT **Enterprise Green IT Readiness:** Introduction, Readiness and Capability, Development of the G-Readiness Framework, Measuring an Organization's G-Readiness **Sustainable IT Services: Creating a Framework for Service Innovation:** Introduction, Factors Driving the Development of Sustainable IT, Sustainable IT Services (SITS), SITS Strategic Framework **Green Enterprises and the Role of IT:** Introduction, Organizational and Enterprise Greening, Information Systems in Greening Enterprises, Greening the Enterprise: IT Usage and Hardware, Inter-organizational Enterprise Activities and Green Issues	15L

Text book:

1) *Harnessing Green IT: Principles and Practices,* San Murugesan, G. R. Ganadharan, Wiley & IEEE.

Additional References:

1) *Green IT,* Deepak Shikarpur, Vishwkarma Publications, 2014
2) *Green Communications: Principles, Concepts and Practice*- Samdanis et al, J. Wiley
3) *Green IT for Sustainable Business Practice: An ISEB Foundation Guide,* Mark G. O'Neill, The Chartered Institute for IT, 2010

Semester II – Practical

Course: USCSP2	Practical of USCS201 + USCS202 + USCS203+USCS204+ USCS205+ USCS206 (Credits : 6, Lectures/Week: 18)	
USCSP201	**Programming with C** 1. Programs to understand the basic data types and I/O. 2. Programs on Operators and Expressions 3. Programs on decision statements. 4. Programs on looping. 5. Programs on arrays. 6. Programs on functions. 7. Programs on structures and unions. 8. Programs on pointers. 9. Programs on string manipulations. 10. Programs on basic file operations.	
USCSP202	**Programming with Python-II** 1. Programs to read and write files. 2. Programs with iterables and iterators. 3. Program to demonstrate exception handling. 4. Program to demonstrate the use of regular expressions. 5. Program to show draw shapes & GUI controls. 6. Program to create server-client and exchange basic information. 7. Program to send email & read contents of URL.	

USCSP203	**Linux** 1. Linux Installation: a. Install your choice of Linux distribution e.g. Ubuntu, Fedora, Debian. b. Try different installation media like CD/DVD, USB Drive to install. c. Customize desktop environment by changing different default options like changing default background, themes, screensavers. 2. a. Screen Resolution: Ascertain the current screen resolution for your desktop. b. Networking: Get the current networking configuration for your desktop. Are you on a wired or a wireless connection? What wireless networks are available, if any? c. Time Settings Change the time zone of your system to (or New York Time if you are currently in Indian time). How does the displayed time change? After noting the time change, change the time zone back to your local time zone. 3. Installing and Removing Software: a. Install gcc package. Verify that it runs, and then remove it. 4. Documentations: a. Finding Info Documentation: From the command line: bring up the info page for the grep command. Bring up the usage section. b. Finding man pages From the command line: Bring up the man page for the ‘ls’ command. Scroll down to the EXAMPLES section. c. Finding man pages by Topic What man pages are available that document file compression? d. Finding man pages by Section From the command line, bring up the man page for the printf library function. Which manual page section are library functions found? e. Command-Line Help List the available options for the mkdir command. How can you do this? 5. Command line operations: a. Install any newpackage on your system b. Remove the package installed c. Find the passwd file in / using find command d. Create a symbolic link to the file you found in last step e. Create an empty file example.txt and move it in /tmp directory using relative pathname. f. Delete the file moved to /tmp in previous step using absolute path. g. Find the location of ls, ps, bash commands. 6. File Operations: a. Explore mounted filesystems on your system. b. What are different ways of exploring mounted filesystems on Linux? c. Archive and backup your home directory or work directory using tar, gzip commands. d. Use dd command to create files and explore different options to dd. e. Use diff command to create diff of two files. f. Use patch command to patch a file. And analyze the patch using diff command again.	

7. Use environment
    a. Which account are you logged in? How do you find out?
    b. Display /etc/shadow file using cat and understand the importance of shadow file. How it’s different than passwd file.
    c. Get you current working directory.
    d. Explore different ways of getting command history, how to run previously executed command without typing it?
    e. Create alias to most commonly used commands like.
8. Linux Editors: vim/emacs
    a. Create,modify, search, navigate a file in editor.
    b. Learn all essential commands like search, search/replace, highlight, show line numbers.
9. Linux Security:
    a. Use of sudo to change user privileges to root
    b. Identify all operations that require sudo privileges
    c. Create a new user and add it to sudo configuration file.
    d. Set password for new user.
    e. Modify the expiration date for new user using password ageing.
    f. Delete newly added user.
10. Network:
    a. Get IP address of your machine using ifconfig.
    b. If IP is not set, then assign an IP address according to your network settings.
    c. Get hostname of your machine.
    d. Use ping to check the network connectivity to remote machines.
    e. Use telnet/ssh to connect to remote machines and learn the difference between the two.
    f. Troubleshooting network using traceroute, ping, route commands.
11. Shell Scripting
    a. Searching with grep: Search for your username in the /etc/passwd file.
    b. Parsing files with awk: Display in a column a unique list of all the shells used for users in /etc/passwd. Which field in /etc/passwd holds the shell (user command interpreter in the manual page)? How do you make a list of unique entries, that is, no repeated entries?
    c. Searching and substituting with sed: Search all instances of the user command interpreter (shell) equal to /bin/false in /etc/passwd and substitute with /bin/bash using sed.
    d. Exit status: write a script which does ls to a non existent file. Display an exit status of the previous command. Now create the file and again display the exit status. In each task send the ls output to /dev/null
    e. Working with files: Write a shell script which will ask user for a directory, create that directory and switch to it and tell the user where you are using pwd command. Now use touch to create some new files followed by displaying the filenames.

f. Environment variables: Write a script which displays all environment variables on the system.

g. Functions: Write a script that asks user for a number (1,2 or 3) which is used to call a function with the number in its name. The function then displays a message with the function number within it, example: “This message is from function number 4.”

h. Arithmetic: Write a script which will work as arithmetic calculator to add, subtract, multiply, divide. The user should pass an argument on the command line a letter (a,s,m or d) and two numbers. If wrong number of arguments are passed then display an error message. Make use of functions to perform operations.

i. Case Statements: Write a script that will be given a month number as the argument and will translate this number into a month name. The result will be printed to stdout.

j. Script Arguments and Usage Information: Write a script that takes exactly one argument, a directory name. The script should print that argument back to standard output. Make sure the script generates a usage message if needed and that it handles errors with a message.

k. Randomness: Create a script that takes a word as an argument from the user, then appends a random number to the word and display it to the user. Put in a check to make sure the user passed in a word, displaying a usage statement if a word was not passed as an argument.

l. Strings: Write a script that will read two strings from the user. The script will perform three operations on the two strings: (1) Use the test command to see if one of the strings is of zero length and if the other is of non-zero length, telling the user of both results. (2) Determine the length of each string and tell the user which is longer or if they are of equal length. (3) Compare the strings to see if they are the same. Let the user know the result.

12. Processes

a. Background and Foreground Jobs: Create a job that writes the date to an output file thrice, with a gap of 60 seconds and 180 seconds. Check whether the job is running and bring it to foreground job. Stop the foreground job and make it run in the background. Finally, kill the background job and verify its status.

b. Scheduling a One-Time Backup: Create job using at to back up files in one directory to another 10 minutes from now.

c. Scheduling Repeated Backups: Set up a cron job to backup the files in one directory to another every day at 10 am. Put the commands in file called mycron.

USCSP204	**Data structures** 1) Implement Linear Search to find an item in a list. 2) Implement binary search to find an item in an ordered list. 3) Implement Sorting Algorithms a. Bubble sort b. Insertion sort c. Quick sort d. Merge Sort 4) Implement use of Sets and various operations on Sets. 5) Implement working of Stacks. (pop method to take the last item added off the stack and a push method to add an item to the stack) 6) Implement Program for a. Infix to Postfix conversion b. Postfix Evaluation 7) Implement the following a. A queue as a list which you add and delete items from. b. A circular queue. (The beginning items of the queue can be reused). 8) Implement Linked list and demonstrate the functionality to add and delete items in the linked list. 9) Implement Binary Tree and its traversals. 10) Recursive implementation of a. Factorial b. Fibonacci c. Tower of Hanoi	
USCSP205	**Calculus** 1. Continuity of functions; Derivative of functions 2. Increasing, decreasing, concave up and concave down functions 3. Relative maxima, relative minima, absolute maxima, absolute minima 4. Newton's method to find approximate solution of an equation 5. Area as a limit and length of a plane curve 6. Numerical integration using Simpson's rule 7. Solution of a first order first degree differential equation, Euler's method 8. Calculation of Partial derivatives of functions 9. Local linear approximation and directional derivatives 10. Maxima and minima of functions of two variables	
USCSP206	**Statistical Methods and Testing of Hypothesis** 1. Problems based on binomial distribution 2. Problems based on normal distribution 3. Property plotting of binomial distribution 4. Property plotting of normal distribution 5. Plotting pdf, cdf, pmf, for discrete and continuous distribution 6. t test, normal test, F test 7. Analysis of Variance 8. Non parametric tests- I 9. Non- Parametric tests – II 10. Post-hoc analysis of one-way analysis	

Evaluation Scheme

I. Internal Exam-25 Marks

(i) Test– 20 Marks

20 marks Test – Duration 40 mins
It will be conducted either using any open source learning management system such as Moodle (Modular object-oriented dynamic learning environment)Or a test based on an equivalent online course on the contents of the concerned course(subject)offered by or build using MOOC (Massive Open Online Course)platform.

(ii) 5 Marks - Active participation in routine class instructional deliveries
Overall conduct as a responsible student, manners, skill in articulation, leadership qualities demonstrated through organizing co-curricular activities, etc.

II. External Examination- 75 Marks

(i) Duration - 2.5 Hours.

(ii) Theory question paper pattern:-

All questions are compulsory.		
Question	Based on	Marks
Q.1	Unit I	20
Q.2	Unit II	20
Q.3	Unit III	20
Q.4	Unit I,II and III	15

- All questions shall be compulsory with internal choice within the questions.
- Each Question may be sub-divided into sub questions as a, b, c, d & e, etc & the allocation of Marks depends on the weightage of the topic.

III. Practical Examination – 300 marks (50 marks x 6 core papers)

- Each core subject carries 50 Marks : 40 marks + 05 marks (journal)+ 05 marks(viva)
- Minimum 75 % practical from each core subjects are required to be completed and written in the journal.

(Certified Journal is compulsory for appearing at the time of Practical Exam)
