DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING

SCHEME OF TEACHING AND EXAMINATION (2017 – 2018)

B.E. (ISE) III SEMESTER

Sl No.	Subject Code	Subject	Credits	Hours/Week			Examination Marks		
				Lecture	Tutorial	Practical	CIE	SEE	Total
1	UMA301C	Engg. Maths III	04	4	0	0	50	50	100
2	UIS322C	Discrete Mathematical Structures	03	3	2	0	50	50	100
3	UIS314C	Computer Organization	04	4	0	0	50	50	100
4	UIS308C	Logic Design	03	3	0	0	50	50	100
5	UIS325C	Data Structures and Algorithms	05	3	2	0	50	50	100
6	UIS312L	Logic Design & Simulation Lab	1.5	0	1	2	50	50	100
7	UIS308L	Data Structures Lab	1.5	0	1	2	50	50	100
8	UIS311L	UNIX & Shell Programming Lab	02	0	2	2	50	50	100
9	UMA300M	Advanced Mathematics-I	--	4	--	--	50	50	100
		Total	**25**	**22**	**7**	**7**	**450**	**450**	**900**

UIS322C: DISCRETE MATHEMATICAL STRUCTURES

3 Credits (3L-0P-2T) (40 Teaching Hours + 26 Tutorial Hours)

Course Objectives : **1)** Comprehend basics of elementary counting techniques.
2) Understand principles of logic and apply laws of logic to check the validity of arguments.
3) Apply operations of sets and principles of inclusion-exclusion in problem solving.
4) Comprehend basics of relations and function and be able to determine their properties.
5) Apply theorems and algorithms from graph theory in problem**.**

Course Outcomes : **1)** To exhibit the ability to apply elementary counting techniques (Using product & sum rule, permutation, combination, the pigeon hole principle, mathematical induction) to solve counting problems.
2) Demonstrate the ability to write logic sentence in terms of predicate, quantifiers and logical connectives and check the validity.
3) To comprehend the basic principles of sets and exhibit the ability to apply the laws of set theory in solving applied problems.
4) Exhibit the ability to apply methods of proof in problem solving and Solve linear recurrence relations by recognizing homogeneity, linearity and constant coefficient.
5) Demonstrate an understanding of relations, functions and be able to determine their properties and Discriminate identify and prove properties of groups and semi groups.
6) Demonstrate the ability to apply theorems and algorithms from graph theory in solving engineering and societal problems.

Mapping (Co-relations) of CO's with PO's

Sl. No.	Course Outcomes	IS-PO-1	IS-PO-2	IS-PO-3	IS-PO-4	IS-PO-5	IS-PO-6	IS-PO-7	IS-PO-8	IS-PO-9	IS-PO-10	IS-PO-11	IS-PO-12
01	To exhibit the ability to apply elementary counting techniques (Using product & sum rule, permutation, combination, the pigeon hole principle, mathematical induction) to solve counting problems.	**3**	**3**	**2**	**2**								
02	Demonstrate the ability to write logic sentence in terms of predicate, quantifiers and logical connectives and check the validity.	2	2	2									
03	To comprehend the basic	3	3	2									

	principles of sets and exhibit the ability to apply the laws of set theory in solving applied problems.												
04	Exhibit the ability to apply methods of proof in problem solving and Solve linear recurrence relations by recognizing homogeneity, linearity and constant coefficient.	3	3	2									
05	Demonstrate an understanding of relations, functions and be able to determine their properties and Discriminate identify and prove properties of groups and semi groups.	3	3	2									
06	Demonstrate the ability to apply theorems and algorithms from graph theory in solving engineering and societal problems.	3	3	2	2	1	3	3					

Syllabus

Subject Title	:	**DISCRETE MATHEMATICAL STRUCTURES**
Subject code	:	**UIS322C**
Semester	:	**3**
Credits with LTP Structure	:	**4 Credits (3L-0P-1T)**
Lecture Hours per Week		**3 Hours**
Tutorial Hours per Week		**2 Hours**
Total Contact Hours	:	**66 (40 Teaching Hours + 26 Tutorial Hours)**

UNIT - I	**16 Hours**	**Teaching Hours**	**Tutorial Hours**
Fundamentals Principles of Counting: The Rules of sum and product, permutations, combinations: the binomial theorem, combinations with repetition, mathematical induction, recursive definitions. **Recurrence relation**: first order linear recurrence relation, the second order linear homogeneous recurrence relation with constant coefficient		**10**	**6**
UNIT - II	**16 Hours**	**Teaching Hours**	**Tutorial Hours**
Fundamentals of Logic: Basic connectives and truth tables, Logical equivalence: the laws of logic, logical implication: rules of inference, the use of		**10**	**6**

quantifiers, quantifiers, definitions and the proofs of theorems. **Set Theory:** Sets and subsets, set operations and the laws of set theory, counting and Venn Diagrams, probability, the axioms of probability.			
UNIT - III	**17 Hours**	**Teaching Hours**	**Tutorial Hours**
Relations and Functions: Cartesian products and relations, functions: plain and one to one, on to functions: sterling numbers of the second kind, special functions, the pigeonhole principle, function composition and inverse functions, properties of relations, computer recognition: zero one matrices and directed graphs, partial order: Hasse diagram, equivalence relations and partitions, lattices. **Semigrops and Groups**: Definition, example and elementary properties, Homomorphism, Isomorphism and cyclic groups, Cosets and lagrange's theorem		**10**	**7**
UNIT - IV	**17 Hours**	**Teaching Hours**	**Tutorial Hours**
An introduction to graph theory: Definitions and examples, subgraphs, complement and graph isomorphism, vertex degree: Euler trails and circuits**.** **Trees:** definitions, properties and examples, rooted trees, trees and sorting weighted trees and prefix codes		**10**	**7**
Text Books:			
Ralph P Grimaldi, **"Discrete and Combinatorial Mathematics-An Applied Introduction",** Pearson Education, 5th Edition.			
Reference Books:			
1. C. L.Lin, "**Elements of Discrete Mathematics**" 2nd Editions 2. Thomas Khoshy "**Discrete Mathematics with applications**" 3. Richard Johasonbangh "**Discrete Mathematics**" 6th Edition 4. Kenneth H rossey "**Discrete Mathematics & etc applications**" 6th edition			

UIS314C: Computer Organization

4 Credits (4-0-0)

Course Outcomes:

After studying this course the student will be able to:

UIS314C-1(CO1)	:	Comprehend the fundamentals of a computing system and its functional units.
UIS314C-2(CO2)	:	Write an Assembly Language Program using various types of instructions and addressing modes.
UIS314C-3(CO3)	:	Perform three approaches to IO i.e. programmed IO, interrupts and DMA.
UIS314C-4(CO3)	:	Comprehend and design various memory organizations.
UIS314C-5(CO5)	:	Implement various arithmetic operations, fast addition, multiplication (BOOTH) and division methods.
UIS314C-6(CO6)	:	Comprehend the design of control unit.

Mapping of Course outcomes with Program Educational Objectives and Program Specific Outcomes.

CO-ID	CO Description	IS-PO-1	IS-PO-2	IS-PO-3	IS-PO-4	IS-PO-5	IS-PO-6	IS-PO-7	IS-PO-8	IS-PO-9	IS-PO-10	IS-PO-11	IS-PO-12
UIS314C-1 (CO1)	Comprehend the fundamentals of computer hardware and software.	1											
UIS314C-2 (CO2)	Write an ALP using all its constructs (PDP11)			3	3	1							
UIS314C-3 (CO3)	Perform three approaches to IO i.e. programmed IO, interrupts and DMA.		1	3			1						1
UIS314C-4 (CO4)	Comprehend and design various memory organizations.		2	3	1								1
UIS314C-5 (CO5)	Implement various arithmetic operations, fast addition, multiplication (BOOTH) and division methods.		3	2									
UIS314C-6 (CO6)	Comprehend the design of control unit.			3	2			1	1				1

Mapping of Course outcomes with Program Specific Outcomes.

CO-ID	CO Description	IS-PSO-1	IS-PSO-2	IS-PSO-3
UIS314C-1(CO1)	Comprehend the fundamentals of computer hardware and software.	1	3	1
UIS314C-2(CO2)	Write a simple ALP using various constructs.	1	3	1
UIS314C-3(CO3)	Perform three approaches to IO i.e. programmed IO, interrupts and DMA.	1	3	1
UIS314C-4(CO4)	Comprehend and design various memory organizations.	1	3	1
UIS314C-5(CO5)	Implement various arithmetic operations, fast addition, multiplication (BOOTH) and division methods.	1	3	1
UIS314C-6(CO6)	Comprehend the design of control unit.	1	3	1

(1-Low, 2-Medium, 3-High Correlations)

Subject Title	:	**COMPUTER ORGANIZATION**
Subject code	:	**UIS314C**
Semester	:	**3**
Credits with LTP Structure	:	**4 Credits (4L-0P-0T)**
Lecture Hours per Week	:	**4 Hours**
Tutorial Hours per Week	:	**00**
Total Contact Hours	:	**52 (52 Teaching Hours + 00 Tutorial Hours)**

UNIT - I	**13 Hours**	**Teaching Hours**	**Tutorial Hours**
Basic Structure of Computer**:** Computer Types. Functional Units, Basic Operational Concepts, Bus Structures, Performance – processor clock, Basic Performance Equation, Clock rate, Performance Measurement. Machine Instructions and Programs**:** Numbers, Arithmetic Operations and Characteristics, Memory Location and Addresses, Memory Operations. Instructions and Instruction Sequencing: Addressing Modes, Assembly language, Basic Input and Output operations, Stacks and Queues, Subroutines.		**13**	**0**

UNIT - II	**13 Hours**	**Teaching Hours**	**Tutorial Hours**
Input/Output organization: Accessing I/O Devices, Interrupts-interrupt hardware, Enabling and disabling interrupts, Handling multiple devices, Controlling device requests, Exceptions, Direct memory access, Buses, Interface circuits, Standard I/O interfaces-USB; Device characteristics,		**13**	**0**

Architecture, Addressing.			
UNIT - III	**13 Hours**	**Teaching Hours**	**Tutorial Hours**
Basic processing unit: Fundamental concepts, Execution of a complete instruction, Multiple bus organization, Hard-wired control, Micro programmed control. Memory system**:** Basic Concepts, Semiconductor RAM Memories, Read Only Memories, Speed, Size and cost, Cache Memories, Mapping Functions.		**13**	**0**
UNIT - IV	**13 Hours**	**Teaching Hours**	**Tutorial Hours**
Basic Arithmetic concepts for ALU: Addition and subtraction of signed numbers, Design of fast adders; Carry-lookahead addition only, Multiplication of positive numbers, Signed operand multiplication, Fast multiplication, Integer division, Floating point numbers and operations.		**13**	**0**
Text Books:			
Carl Hamacher, Zvonko Vranesic, Safwat Zaky, "**Computer Organization**", 5th edition, TMH, 2002.			
Reference Books:			
William Stallings, **"Computer Organization and Architecture",** 7th edition,PHI,2006			

UIS308C: Logic Design

(3-0-0) (3 Credits, 40 Hours)

Course Objectives:

1. To acquire the basic knowledge of digital logic levels and application of knowledge to understand digital electronics circuits.
2. To prepare students to perform the analysis and design of various digital electronic circuits.

Course Outcomes:

1. Comprehend the fundamentals of Digital Logic and Analog Systems.
2. Design basic Digital Systems using Boolean algebra.
3. Analyze and Design digital solutions for a given problem.
4. Implement digital solutions for a given problem.
5. Analyze and Synthesize digital systems using Sequential Systems.

Mapping (Co-relations) of COs with POs

S.N	Course Outcomes	IS –PO -1	IS –PO-2	IS –PO-3	IS -PO-4	IS -PO-5	IS -PO-6	IS -PO-7	IS -PO-8	IS -PO-9	IS -PO_10	IS -PO_11	IS -PO_12
CO-1	Comprehend the fundamentals of Digital Logic and Analog Systems.	2											
CO-2	Design basic Digital Systems using Boolean algebra.			3									
CO-3	Analyze and Design digital solutions for a given problem.		3	3									
CO-4	Implement digital solutions for a given problem.			3									
CO-5	Analyze and Synthesize digital systems using Sequential Systems.			3	2								

Syllabus

Academic Year	**:**	**2018-19**
Semester	**:**	**3rd**
Subject	**:**	**Logic Design**
Subject Code	**:**	**UIS308C**
Credits	**:**	**3**
Teaching Hours	**:**	**40**
UNIT-I		**10 Hours**

Boolean Algebra and Combinational Networks: Definition of Boolean algebra, Boolean formulas and functions, Canonical Formulas, Manipulations of Boolean formulas

Gates and Combinational networks, Incomplete Boolean functions and Don't care conditions, Additional Boolean operations and Gates

UNIT-2	**10 Hours**

Simplifications of Boolean Expressions:
Formulations of simplification problem, Prime implicants and Irredundant disjunctive expressions, Prime implicates and Irredundant conjunctive expressions, Karnaugh maps, Using Karnaugh maps to obtain minimal expressions for complete Boolean functions, Minimal expressions of incomplete Boolean functions

The Quine-McCluskey method of generating Prime implicants and Prime implicates, Decimal method for obtaining prime implicants, Variable-Entered Karnaugh maps.

UNIT-3	**10 Hours**

Logic Design with MSI Components and Programmable Logic Devices:
Binary adders and subtracters, Decimal adders, Comparators, Decoders, Multilpexers

Programmable logic devices (PLDs), Programmable read only memories (PROMs), Programmable logic arrays (PLAs), Programmable array logics (PALs)

UNIT-4	**10Hours**

Flip-Flops and Simple Flip-Flop Applications:
The basic Bistable element, Lathes, Master-Slave flip-flops (Pulse-Triggered flip-flops), Edge triggered flop-flops, Characteristic equations, Registers

Counters, Design of Synchronous Counters, **Synchronous sequential networks:** Structure and operation of clocked synchronous sequential networks, Analysis of clocked synchronous sequential networks

Text Book	**:**	1. Donald D. Givone, "Digital Principles and Design", McGraw Hill Edition 2002 (Chapter 3: 3.1, 3.4, 3.5, 3.6, 3.7, 3.8, 3.9, Chapter 4: 4.1, 4.2, 4.3, 4.4, 4.5, 4.6, 4.8, 4.11, 4.14, Chapter 5: 5.1, 5.2, 5.3, 5.4, 5.6, 5.7, 5.8, 5.9, 5.10, Chapter 6: 6.1, 6.2, 6.4, 6.5, 6.6, 6.7, 6.8, 6.9, Chapter 7: 7.1, 7.2)
Reference Books	**:**	1. Leach and Malvino, "**Digital Principles and Applications**", TMH, New Delhi, 2002. 2. Yarbrough J. M, "**Digital logic- Applications and Design**, Thomson Learning, New Delhi, 2001.

UIS325C: Data Structures and Algorithms
(4-0-1)(5 Credits, 52 Hours, 2 Tutorials)

Course Objectives :

1) Learn to design software using abstract data types , such as Stack ,Queue, Linked list and trees.
2) Learn to use recursion in program design.
3) Learn to implement ADT's in alternate ways.
4) Learn to quantitatively evaluate alternate implementations and explain the tradeoffs involved.

Course Outcomes :

1) Comprehend advanced C programming concepts and demonstrate the ability to write C programs using pointers, structures, unions.
2) Develop algorithms to simulate various linear data structures like stack, queue and linked lists.
3) Comprehend the concepts of recursion and write C programs using recursion.
4) Develop algorithms to simulate non linear data structures like Binary tree, Binary search tree.
5) Implement some applications using linear and non linear data structures.

Mapping (Co-relations) of CO's with PO's

S.No.	Course Outcomes	IS –PO -1	IS –PO-2	IS –PO-3	IS -PO-4	IS -PO-5	IS -PO-6	IS -PO-7	IS -PO-8	IS -PO-9	IS –PO-10	IS –PO-11	IS –PO-12	Total HML's
CO-1	Comprehend advanced C programming concepts and demonstrate the ability to write C programs using pointers, structures, unions.	3	3	3										9
CO-2	Develop algorithms to simulate various linear data structures like stack, queue and linked lists.	3	3			3		2	3	2				16
CO-3	Comprehend the concepts of recursion and write C programs using recursion.	3	3	3										9
CO-4	Develop algorithms to simulate non linear data structures like Binary tree, Binary search tree.	3	3			3		2	3	2				16

CO-5	Implement some applications using linear and non linear data structures.		3	3				3	3	2				14

Syllabus

Academic Year	**:**	**2018 – 19**
Semester	**:**	**03**
Subject	**:**	**Data Structures and Algorithms**
Subject Code	**:**	**UIS325C**
Credits	**:**	**05(4L-0P-1T)**
Teaching Hours	**:**	**52**

Unit –I — **13 Hours**

Pointers: Introduction, Pointers for Inter-Function Communication, Pointers to Pointers, Compatibility, Lvalue and Rvalue,

Pointer Applications: Arrays and Pointers, Pointer Arithmetic and Arrays, Passing an array to a Function, Memory Allocation Functions, Array of Pointers.

Enumerated, Structure and Union Types: The Type Definition, Enumerated Types, Structure, Unions.

Unit –II — **13 Hours**

The stack:

Definition and Examples: Primitive operations, An Example, The stack as an Abstract data type. , Representing Stacks in C: Implementing pop operation, Testing for exceptional conditions, Implementing the push operations. , An Example- Infix, Postfix and Prefix: Evaluating a postfix expression, Program to evaluate a postfix expression, Limitations of the program, Converting an expression from Infix to Postfix, Program to convert an expression from Infix to Postfix.

Recursion:

Recursive definition and processes: The factorial function, Multiplication of Natural numbers, The Fibonacci sequence, The Binary search, Properties of recursive definitions or Algorithms. , Recursion in C: Factorial in C, The Fibonacci Numbers in C, The binary search in C, Recursive chains, Recursive definitions of algebraic expressions. , writing recursive programs: The Towers of Hanoi Problem.

Unit -III — **13 Hours**

Queues:

The queue and its sequential representation: The queue as an abstract data type, C implementation of queues, The insert operation, The priority queue, Array implementation of a priority queue**.**

Lists:

Linked lists: Inserting and removing nodes from a list, Linked implementation of stacks, The getnode and freenode operations, Linked implementation of queues, The linked list as a data structure, Examples of list operations, List implementation of priority queues, Header Nodes.

Lists in C: Array implementation of lists, Limitations of the array implementation, Allocating

and freeing dynamic variables, Linked lists using dynamic variables, Queues as lists in C, Examples of list operations in C, Non integer and nonhomogeneous lists, Comparing the dynamic and array implementation of lists, Implementing Header Nodes.

Simulation using linked lists: Applications of queues, priority queues and linked lists in simulation.

Unit-IV	**13 Hours**

Other list structures: Circular lists, The stack as a circular list, The queue as a circular list, Primitive operations on circular lists. Applications of circular list – the Josephus problem, Addition of long positive integers using circular lists**.**

Trees:

Binary trees: Basics, Operation on Binary trees, Applications of Binary tree.

Binary tree representation: Node representations of Binary trees, Internal & external nodes, Implicit array representation of Binary trees. Choosing a Binary tree representation, Binary tree traversal in c.. Traversal using father field, Heterogeneous Binary trees. ***Trees and their applications***: C representation of trees, Tree traversals, General expressions as trees, Evaluating an expression tree, Constructing tree.

Text Books	**:**	**1)** Behrouz A. Forouzan, Richard F. Gilberg, "**A Structured Programming Approach Using C**", Third Edition, Thomson Brooks/Cole . **2)** Aaron M. Tennenbaum, Yedidyah Langsam and Moshe J. Augenstein,"**Data structure using C**", Pearson Education/PHI 2006.
Reference Books	**:**	**1)** Behrouz A. Forouzan and Richard F. Gilberg, Thomson, "**Computer Science A structured Programming Approach using C",** II edition, 2003. **2)** Richard F. Gilberg and Behrouz, Data structures "**A pseudo code approach with C** ", Thomson, 2005. **3)** Robert Kruse and Breuse Leung, "**Data structures and program Design in C**", PEARSON Education, 2007

UIS312L: LOGIC DESIGN AND SIMULATION LABORATORY

1.5 Credits (0 – 1 - 2)

List of Assignments

PART A: Hardware Experiments

1. Realization of a given Boolean Expression using MEV method.
2. Design and implementation of BCD to Excess-3 using 4-bit Adder Chip and Logic Gates.
3. Design and implementation of full adder using 3:8 Decoders (74138).
4. Design and implementation of full substractor using 8:1 multiplexer (74154).
5. Design JK master/slave flip-flop using NAND gates.
6. Design and implementation of 3 bit Mod-n synchronous counter using JK flip-flops (7476) (where n<=8).
7. Design and implementation of Ring counter and Johnson counter using 4-bit shift register.
8. Design and implementation of an Asynchronous Counter using a Decade Counter IC to count up from 0 to n (n<9). Display the count value on 7-segment LED display using BCD to 7-segment code converter IC.

PART B: Software Programs

Simulation of the following using ALTERA MAX PLUS2 software:

1. Given Boolean expression.
2. BCD to Excess-3 conversion.
3. Full adder.
4. 8:1 Multiplexer.
5. JK master/slave flip-flop
6. 3 bit Mod-n synchronous counter using JK flip-flops
7. Ring counter and Johnson counter.

General remarks:

1. Lab schedule: 3hrs/week for each student (1 hr tutorial, 2 hrs- program execution).
2. Student should complete all the lab assignments.
3. Evaluation CIE 50 marks:
   Lab assignments : 30 marks
   Lab CIE : 20 marks

UIS308L: DATA STRUCTURE LAB

1.5 CREDITS (0-1-2)

1) Write a C program to compute roots of a quadratic equation. Use pointers to pass data from a read function, pass both values and pointers to a compute function, and finally pass the values to a print function.

2) Write a C program that uses an array of student structures to answer inquiries. Using menu driven user interface, provide inquiries as below. The first menu option accepts student information such as name, USN, marks for 5 subjects, second option report a student USN, marks, average & grade based on absolute scale of 90% for A, 80% for B, 70% for C, 60% for D and score below 60% is an F & third option provides all data for requested student.

3) Write a C program to implement integer stack ADT using arrays.

4) Write a C Program to convert and print a given valid parenthesized infix arithmetic expression to postfix expression. The expression consists of single character operands and the binary operators + (plus), - (minus), * (multiply) and / (divide).

5) Write recursive C Program for solving the Towers of Hanoi problem.

6) Write a C program to implement integer queue ADT using arrays.

7) Write a C program using dynamic variables and pointers, to construct a singly linked list consisting of the following information in each node: student id (integer), student name (character string) and semester (integer). The operations to be supported are:

   a. The insertion operation
      i. At the front of a list
      ii. At the back of the list
      iii. At any position in the list

   b. Deleting a node based on student id. If the specified node is not present in the list an error message should be displayed. Both the options should be demonstrated.
   c. Searching a node based on student id and update the information content. If the specified node is not present in the list an error message should be displayed. Both situations should be displayed.
   d. Displaying all the nodes in the list.

8) Write a C program to implement integer queue ADT using singly linked list.

9) Write a C program to construct a circular linked list. Implement insertion and deletion operations on the circular linked list.

10) Write a C program to construct a binary tree & implement tree traversal methods.

General remarks:

4. Lab schedule: 3hrs/week for each student (1 hr tutorial, 2 hrs- program execution).
5. Student should complete all the lab assignments.
6. Evaluation CIE 50 marks:
   Lab assignments : 30 marks
   Lab CIE : 20 marks

UIS311L: UNIX AND SHELL PROGRAMMING LABORATORY
2.0 CREDITS (0-2-2)

WEEK I:
Brief history of UNIX, UNIX components, commands in Unix, some basic commands, command substitution, giving multiple command, aliases.

WEEK II:
Unix files, categories of files, hidden files, the file system, pathnames, the home directory, directory commands, dot and double dot filenames, file related commands, wild cards, displaying the content of the file, printing the files, comparing the files.

WEEK III:
Ownership of files, file attributes, the is command, file command, chmod command; chown command, chgrp command, times associated with a file, umask command.

WEEK IV:
Standard I/O, Redirection, pipes and filters: standard I/O, redirection, pipe and pipeline, mixing input from standard i/p and file, filler, the tee command, terminal and trash files, sample database file, handling columns and fields, sort, *tr* and uniq commands.

WEEK V:
Vi editor, editing with vi, moving the cursor, editing, copying and moving with text, pattern searching, repeating the last editor command, undoing commands, joining lines, writing selected lined onto a file, using shell from vi.

WEEK VI:
Regular expression, grep family of commands and the sed, regular expressions, the grep family, egrep command, fgrep command. stream editor-sed. The process: the meaning, the parent and the child process, types of processes, internal and external commap.ds, ps command, process creation, nohup command, nice command, time command, command history.

WEEKVIII:
Shell variables, the export command, .profile file, the first shell script, the read command, positional parameters, the $? Variable, Set command, exit command. Branching control structures.

WEEK IX:
Loop control structures, the continue and break statements, the expr command, real arithmetic, the here documents, the sleep command, debugging scripts.

WEEK X:
The script commands: the eval command, exec command, more scripts.

Text Book:

1. M. G. Venkatesh Murthy "Introduction UNIX and Shell Programming". Pearson publication, 2005.

General Remarks:

1. Lab schedule: 3hrs/week for each student (2 hr tutorial, 2 hrs- program execution).
2. Student should complete all the lab assignments.
3. Marks Evaluation:
   Lab assignments : 30 marks
   Lab CIE : 20 marks