SEMESTER 1

SEMESTER – I

20MCA101	MATHEMATICAL FOUNDATIONS FOR COMPUTING	CATEGORY	L	T	P	CREDIT
		GENERAL	3	1	0	4

Preamble: This course introduces students to some basic mathematical ideas and tools which are at the core of MCA course. It introduces the concepts of graph theory, set theory and statistics.

Prerequisite: A basic course in set theory and statistics.

Course Outcomes: After the completion of the course the student will be able to

CO 1	Understand mathematical reasoning in order to read, comprehend and construct mathematical arguments
CO 2	Count or enumerate objects and solve counting problems and analyze algorithms
CO 3	Solve problems in almost every conceivable discipline using graph models
CO 4	Solve the linear system of equations and Calculate the eigen values and eigen vectors of matrices.
CO 5	Apply the principles of correlation and regression in practical problems.

Mapping of course outcomes with program outcomes

	PO 1	**PO 2**	**PO 3**	**PO 4**	**PO 5**	**PO 6**	**PO 7**	**PO 8**	**PO 9**	**PO 10**	**PO 11**	**PO 12**
CO 1	3	3	3	3			3					
CO 2	3	3	3	3			3					
CO 3	3	3	3	3			3					
CO 4	3	3	3	3			3					
CO 5	3	3	3	3			3					

Assessment Pattern

Bloom's Category	**Continuous Assessment Tests**		**End Semester Examination**
	1	**2**	
Remember(K1)	10	10	10
Understand(K2)	20	20	20
Apply(K3)	20	20	30
Analyse(K4)			
Evaluate(K5)			
Create(K6)			

Mark distribution

Total Marks	**CIE**	**ESE**	**ESE Duration**
100	40	60	3 hours

Continuous Internal Evaluation Pattern:

Attendance	: 8 marks
Continuous Assessment Test (2 numbers)	: 20 marks
Assignment/Quiz/Course project	: 12 marks

End Semester Examination Pattern: There will be two parts; Part A and Part B. Part A contains 10 compulsory short answer questions, 2 from each module. Each question caries 3 marks. Part B contains 2 questions from each module of which student should answer any one.Each question can have maximum 2 sub-divisions and carry 6 marks.

Course Level Assessment Questions

Course Outcome 1 (CO1):

1. Define equivalence relation with suitable example. (K1)

2. Write Warshall's algorithm. Use to find the transitive closure of the relation

   {(1,3), (3,2), (2,4), (3,1), (4,1) on (1,2,3,4)} (K2)

3. Let f,g:R| |R be defined by f(x)= x+1, g(x)= $2x^2$+3, find fog and gof. Is fog = gof ? (K3)

Course Outcome 2 (CO2)

1. Solve the linear Diophantine equation 24x+138y = 18 (K5)
2. Find the GCD (12378,3054) (K3)
3. Solve a_{n+2}- $4a_{n+1}$+$3a_n$= -200,n | |0 given that a_o= 3000, a_1=3300 (K5)

Course Outcome 3(CO3):

1. Define Hamilton cycle and Euler circuit with example. (K1)

2. Show that $K_{3,3}$ is non-planar. Define planar graph. State Kuratowski's theorem. (K4)

3. Prove that a connected graph G is an Euler graph if all vertices of G are of even degree. (K4)

Course Outcome 4 (CO4):

1. Find the rank of the matrix $\begin{bmatrix} 0 & 3 & 4 \\ -3 & 0 & -5 \\ -4 & 5 & 0 \end{bmatrix}$ (K3)
2. Find the Eigen values and Eigen vectors of $\begin{bmatrix} 4 & 2 & -2 \\ 2 & 5 & 0 \\ -2 & 0 & 3 \end{bmatrix}$ (K3)
3. Find out what type of conic sections the quadratic form $Q = 17x_1^2 - 30x_1x_2 + 17x_2^2 = 128$ represents and transform it into principal axes form (K3)

Course Outcome 5 (CO5):

1. State the principle of least squares. (K1)
2. Fit a parabola by the method of least squares, to the following data. (K3)

x:	1	2	3	4	5
y:	5	12	26	60	97

3. Compute the correlation coefficient from the following data. (K3)

x:	77	54	27	52	14	35	90	25	96	60
y:	35	58	60	40	50	40	35	56	34	42

Syllabus

Module 1

Sets, Set Operations, Relations, Classification of relations, Equivalence Relations, Closures of Relations, Matrix Representation of Relations, Partial Ordering, n-ary Relations, Functions.

Module 2

Division Algorithm, GCD, Primes, Euclidean Algorithm, Congruences, Properties of Congruences, Solutions of Linear Congruences.

First Order Linear Recurrence Relation, Second Order Linear Homogeneous Recurrence Relations with Constant coefficients, Non Homogeneous Recurrence Relation.

Module 3

Graphs and Graph Models, Graph Terminology and Special Types of Graphs, Representing Graphs and Graph Isomorphism, Connectivity, Euler and Hamilton Paths, Shortest-Path Problems, Planar Graphs, Directed Graph, Multigraph, Connected graph, Euler circuit and trail, Planar and Non-planar Graphs.

Module 4

Linear system of equations, coefficient matrix, augmented matrix, Gauss elimination method and back substitution, elementary row operations, row equivalent systems, Gauss elimination- three possible cases, Row Echelon form and information from it, Linear independence- rank of a matrix. Solution of linear system, fundamental theorem of non- homogeneous linear system (without proof). Homogeneous linear system (theory only), Matrix eigen value problem- determination of eigen values and eigen vectors, Basis of eigen vectors- diagonalization of matrix- Quadratic form- principle axis theorem (without proof).

Module 5

Bivariate data – Scatter Diagram – Interpretation of the nature and degree of relation using scattered diagram - Curve fitting – Principle of least squares – fitting a straight line – fitting a parabola – linear correlation and regression – Karl's Pearson's Coefficient of Correlation – Spearman's rank correlation coefficient (problems based on the formula).

Text Books

1. David M. Burton, "Elementary Number Theory", McGraw-Hill, 7th Edition (2012).

2. Ralph P Grimaldi, "Discrete and Computational Mathematics: An applied introduction", Pearson Education, 5th Edition, (2007).

3. Erwin Kreyszig, "Advanced Engineering Mathematics", 10th ed., Wiley.
4. Gupta S.C and Kapoor V .K, "Fundamentals of Mathematical Statistics", Sultan Chand and Sons 11th edition.

Reference Books

1. C. Liu, “Elements of Discrete Mathematics: A Computer Oriented Approach”, McGraw-Hill, 4th Edition (2012).
2. Jean-Paul Tremblay ,“Discrete Mathematical Structures with applications to Computer science”, ”, McGraw-Hill, 1st Edition (2001).
3. Kenneth H. Rosen , ”Discrete mathematics and its applications”, McGraw-Hill, (7th Edition), (Smartbook available).
4. Marty Lewinter, Jeanine Meyer, “Elementary Number Theory with Programming”, Wiley- Blackwell (2015).
5. David S. Moore and George P. McCabe, “Introduction to practice of statistics”, W.H. Freeman & Company, 5th Edition (2005).
6. Douglas C. Montgomery and George C. Runger, “Applied Statistics and Probability for Engineers”, Wiley India, 5th Edition (2012).
7. Veerarajan T, “Probability and Random Process”, 3rd Edition, Tata McGraw-Hill (2002)
8. G. Jay Kerns, “Introduction to Probability and Statistics Using R”, Chapman & Hall (2010).
9. B.S Grewal. Higher Engineering Mathematics, Khanna Publishers, New Delhi.

Web Resources

1. Probability and statistics EBook
   http://wiki.stat.ucla.edu/socr/index.php/EBook
2. https://www.openintro.org/stat/textbook.php
3. http://www.math.uah.edu/stat/index.html
4. Statistics Online Computational Resource
   http://www.socr.ucla.edu/

Course Contents and Lecture Schedule

Topic	No. of lectures
Module 1	9 hrs.
Sets, Set Operations	2
Relations, Classification of relations, Equivalence Relations	2
Closures of Relations, Matrix Representation of Relations, Partial Ordering, n-ary Relations	3
Functions	2

Module 2	9 hrs.
Division Algorithm, GCD, Primes, Euclidean Algorithm	2
Congruences, Properties of Congruences, Solutions of Linear Congruences	2
First Order Linear Recurrence Relation	1
Second Order Linear homogeneous Recurrence Relations with Constant coefficients	2
Non Homogeneous Recurrence Relation	2
Module 3	8 hrs.
Graphs and Graph Models, Graph Terminology and Special Types of Graphs	1
Representing Graphs and Graph Isomorphism, Connectivity	2
Euler and Hamilton Paths, Shortest-Path Problems, Planar Graphs	2
Directed Graph, Multigraph, Connected graph	1
Euler circuit and trail, Planar and Non-Planar Graphs	2
Module 4	11 hrs.
Linear system of equations, coefficient matrix, augmented matrix, Gauss elimination method and back substitution, elementary row operations, row equivalent systems	2
Gauss elimination- three possible cases, Row Echelon form and information from it	2
Linear independence- rank of a matrix. Solution of linear system, fundamental theorem of non- homogeneous linear system (without proof). Homogeneous linear system (theory only), fundamental theorem of non- homogeneous linear system (without proof). Homogeneous linear system (theory only)	3
Matrix eigen value problem- determination of eigen values and eigen vectors, Basis of eigen vectors	2
diagonalization of matrix, Quadratic form-principle axis theorem (without proof).	2
Module 5	8 hrs.
Bivariate data – Scatter Diagram – Interpretation of the nature and degree of relation using scattered diagram	2
Curve fitting – Principle of least squares – fitting a straight line – fitting a parabola	2
linear correlation and regression – Karl's Pearson's Coefficient of Correlation	2
Spearman's rank correlation coefficient	2

20MCA103	DIGITAL FUNDAMENTALS & COMPUTER ARCHITECTURE	CATEGORY	L	T	P	CREDIT
		GENERAL	3	1	0	4

Preamble:

The primary aim of this course is to understand the fundamentals behind the digital logic design and gain the experience to design digital circuits and systems. Students should also acquire some understanding and appreciation of a computer system's functional components, their characteristics, performance and interactions. They need to understand the computer architecture in order to make best use of the software tools and computer languages they use to create programs.

Prerequisite: NIL

Course Outcomes: After the completion of the course the student will be able to

CO 1	Apply the basics of digital electronics to design and realize simple combinational logic circuits
CO 2	Apply the digital electronics principles to design sequential logic circuits.
CO 3	Understand the different design features of computer architecture, Five key components of a computer, processor and memory making technologies, addressing modes & instruction formats.
CO 4	Understand Processor logic design conventions and data path, pipelining and hazards, I/O organization, Interrupts and direct memory access
CO 5	Understand and different types of memories - RAM, ROM, Cache memory, virtual memory etc. Apply the different memory design techniques.
CO 6	Understand the concept of single board computers like Arduino, Raspberry Pi etc. and apply the same in practical applications.

Mapping of course outcomes with program outcomes

	PO 1	**PO 2**	**PO 3**	**PO 4**	**PO 5**	**PO 6**	**PO 7**	**PO 8**	**PO 9**	**PO 10**	**PO 11**	**PO 12**
CO 1	3	3	2	1	-	-	1	-	-	-	-	-
CO 2	3	3	2	1	-	-	1	-	-	-	-	-
CO 3	1	1	-	1	-	-	1	-	-	-	-	-
CO 4	1	1	-	-	-	-	1	-	-	-	-	-
CO 5	2	2	1	1	-	-	1	-	-	-	-	-
CO 6	1	1	2	-	2	-	2	2	2	-	2	2

Assessment Pattern

Bloom's Category	Continuous Assessment Tests		End Semester Examination
	1	**2**	
Remember	10	10	20
Understand	20	20	20
Apply	20	20	20
Analyse			
Evaluate			
Create			

Mark distribution

Total Marks	CIE	ESE	ESE Duration
100	40	60	3 hours

Continuous Internal Evaluation Pattern:

Attendance	: 8 marks
Continuous Assessment Test (2 numbers)	: 20 marks
Assignment/Quiz/Course project	: 12 marks

End Semester Examination Pattern: There will be two parts; Part A and Part B. Part A contain 10 questions with 2 questions from each module, having 3 marks for each question. Students should answer all questions. Part B contains 2 questions from each module of which student should answer any one. Each question can have maximum 2 sub-divisions and carry 6 marks.

Course Level Assessment Questions

Course Outcome 1 (CO1):

1. Minimize the Boolean Expression f(A,B,C) = Σm(1,3,5,6,7) using K-map.

2. Convert the decimal number 3.257 * 10^4 into single precision floating point binary representation

3. Express -31 in sign magnitude,1's complement and 2's complement notations

Course Outcome 2 (CO2)

1. Explain J-K flipflop with its truth table

2. Design an asynchronous decade counter.

3. Describe the working of a Parallel in Serial Out register.

Course Outcome 3 (CO3):

1. Describe the key components of a computer.

2. Define addressing mode. List 5 addressing modes with examples.

3. Differentiate between fixed length encoding and variable length encoding.

Course Outcome 4 (CO4):

1. Define pipeline, describe how pipeline improves the performance of the machine.

3. List different types of pipeline hazards with examples.

2. Explain how interrupts from multiple devices handled?

Course Outcome 5 (CO5):

1. Illustrate different cache mapping techniques with neat diagrams.

2. Discuss about Read Only Memories

3. Design 2M*32 memory module using 512K *8 static memory chips.

Course Outcome 6 (CO6):

No questions for university examination, for internal assessments practical assignment for configuring a PC / arduino or raspberry and programming assignments using HDL like Verilog or VHDL can be given.

Model Question paper

Reg No.:_______________		Name:___________________________________	
APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY FIRST SEMESTER M.C.A.DEGREE EXAMINATION, MODEL QUESTION PAPER			
Course Code: 20MCA103			
Course Name: DIGITAL FUNDAMENTALS & COMPUTER ARCHITECTURE			
Max. Marks: 60			Duration: 3 Hours
PART A			
		Answer all questions, each carries 3 marks.	Marks
1		Represent +45,-45 in 1's complement and 2's complement form.	(3)
2		Implement a full adder using 8:1 MUX	(3)
3		How could you convert RS flip flop to D flip flop?	(3)
4		What is meant by modulus of a counter? Realize a mod-8 synchronous counter.	(3)
5		Suppose we have two implementations of the same instruction set architecture. Computer A has a clock cycle time of 250 ps and a CPI of 2.0 for some program, and computer B has a clock cycle time of 500 ps and a CPI of 1.2 for the same program. Which computer is faster for this program and by how much?	(3)
6		Describe about little endian and big endian byte ordering.	(3)
7		Explain 4 stage pipelining with a diagram.	(3)
8		Differentiate between memory mapped I/O and Isolated I/O	(3)
9		What is static RAM ?	(3)
10		Define temporal locality and spatial locality.	(3)
PART B			
Answer any one question from each module. Each question carries 6 marks.			
Module I			
11		Explain about single precision floating point representation with an example	(6)
OR			
12		Minimize the Boolean expression f(A,B,C,D)=Σm(1,5,6,7,9,15)+d(2,3,11,13) using Karnaugh map and realize it using NAND gates.	(6)

Module II			
13		Demonstrate the working of a JK flip flop. How does it eliminate the invalid condition in SR flip flop? List out its applications.	(6)
OR			
14		Design a mod-12 asynchronous counter.	(6)
Module III			
15		Explain the five classic components of a computer with diagram.	(6)
OR			
16		Describe the code sequence of C=A+B in different types of instruction set architecture.	(6)
Module IV			
17		Draw a single datapath representation for memory instructions and R-type instructions	(6)
OR			
18		What is Direct Memory Access? Explain two types of bus arbitration schemes	(6)
Module V			
19		Elaborate the various cache memory mapping techniques with an example for each.	(6)
OR			
20		Explain the internal organization of memory chips and design a 1K*1 memory chip using decoder.	(6)

Syllabus

Module I (11 Hours)

Representation of signed numbers – 1's complement and 2’s complement ,Logic gates – AND - OR – NOT - NAND- NOR - XOR , Boolean algebra - Basic laws and theorems , Boolean functions - truth table, Standard forms of Boolean Expressions – Sum of Products and Product of Sums - minimization of Boolean function using Karnaugh map method - Realization using logic gates, Floating point numbers
Combinational Circuits - Half adder - Full Adder- Decoder -Encoder- Multiplexer – Demultiplexer

Module II (10 Hours)

Sequential circuit - Clocking, Flip flops - SR – JK- D -T flip flops, Counters - Synchronous and asynchronous counters - UP/DOWN counters , Registers - Serial in serial out - Serial in parallel out - Parallel in serial out - Parallel in parallel out registers

A practical assignments may be given in configuring a PC / configuring arduino - Implementing simple programs for blinking an LED - Input from an external switch - fading an LED - serial monitor and debugging / installing & configuring Raspberry pi.

Module III (10 Hours)

Computer abstractions and technology - Introduction, Computer architecture -8 Design features, Application program - layers of abstraction, Five key components of a computer, Technologies for building processors and memory, Performance, Instruction set principles – Introduction, Classifying instruction set architectures, Memory addressing, Encoding an instruction set.

Module IV (9 Hours)

The Processor - Introduction, Logic design conventions, Building a datapath, A simple implementation scheme, An overview of pipelining - Pipelined datapath and control - Structural hazards - Data hazards - Control hazards

I/O organization - Accessing I/O devices, interrupts - handling multiple devices, Direct memory access

Programming assignments may be given in any HDL like Verilog or VHDL to create gate level/ Dataflow/Behavioural level models of gates, multiplexers, adders, flip-flops, registers etc. No detailed teaching of HDL is necessary. The students can be given a basic tutorial write up on gate level modelling.

Module V (8 Hours)

The Memory System – basic concepts, semiconductor RAM memories - organization – static and dynamic RAM, Structure of larger memories, semiconductor ROM memories, Speed, Size and cost ,Cache memory – mapping functions – replacement algorithms , Virtual memory – paging and segmentation.

Text Books

1. Floyd, "***Digital Fundamentals***", Pearson Education, 10th Edition (2011).(Module 1 & 2)
2. J. Hennessy and D. Patterson, "***Computer Organization and Design: The Hardware/Software Interface***", 5th Edition. (Module 3 & 4)
3. J. Hennessy and D. Patterson, "***Computer Architecture, A quantitative approach***", 5th Edition. (Module 3)
4. Hamacher, Vranesic & Zaky, "***Computer Organization***" (5th Ed), McGraw Hill. (Module 4 & 5)

References

1. William Stallings, "***Computer Organization and Architecture: Designing for Performance***", Pearson, 9/e, 2013.
2. R.P.Jain ,"***Modern Digital Electronics***", McGraw Hill.,Fourth Edition,2009
3. Mano, "***Digital Design : With an Introduction to Verilog HDL***", Pearson Education, 5th Edition (2014)

Course Contents and Lecture Schedule

No	Topic	No. of Lectures
	Module 1	**11**
1	Representation of signed numbers – 1's complement and 2's complement, Logic gates - AND, OR, NOT, NAND, NOR, XOR	2
1.1	Boolean algebra - Basic laws and theorems, Boolean functions - truth table.	2
1.2	Standard forms of Boolean Expressions – Sum of Products and Product of Sums - minimization of Boolean function using Karnaugh map method - Realization using logic gates.	2
1.3	Floating point numbers	1
1.4	Combinational Circuits - Half adder - Full Adder	2
1.5	Decoder – Encoder - Multiplexers – Demultiplexers	2
	Module 2	**10**
2.1	Sequential circuit - Clocking, Flip flops -RS – JK- D -T flip flops	3
2.2	Counters - Synchronous and asynchronous counters - UP/DOWN counters.	3
2.3	Registers - Serial in serial out - Serial in parallel out - Parallel in serial out - Parallel in parallel out registers	2
2.4	Introduction to arduino and raspberry pi	2
	Module 3	**10**
3.1	Computer abstractions and technology - Introduction, Computer architecture	4
3.2	Technologies for building processors and memory, Performance, instruction	4
3.3	Classifying instruction set architectures, Memory addressing, Encoding an	2
	Module 4	**9**
4.1	The Processor - Introduction, Logic design conventions, Building a datapath, A simple implementation scheme.	3
4.2	An Overview of pipelining - Pipelined datapath and control - Structural hazards - Data hazards - Control hazards	3
4.3	I/O organization - Accessing I/O devices, Interrupts - Handling multiple devices- Direct memory access	3

No	Topic	No. of Lectures
	Module 5	**8**
5.1	The memory system – basic concepts, semiconductor RAM memories, organization	2
5.2	Static and dynamic RAM, Structure of larger memories, semiconductor ROM memories, Speed, size and cost	2
5.3	Cache memory – mapping functions – replacement algorithms,	2
5.4	Virtual memory – paging and segmentation.	2

20MCA105	ADVANCED DATA STRUCTURES	CATEGORY	L	T	P	CREDIT
		GENERAL	3	1	0	4

Preamble: A graduate course in Computer Applications should give due exposure to the recent developments. Since Data structures is a central pillar of any program on Computer Science/ Applications, this course is designed to build upon the knowledge acquired at the undergraduate level and familiarise students with a bunch of modern data structures which are quite useful to solve, in the most effective manner, the modern, real life problems.

Prerequisite: Basic Data Structures

Course Outcomes: After the completion of the course the student will be able to

CO 1	Remember the Basic Data Structures and understand the Set Data Structure and its implementation.
CO 2	Understand Advanced Tree Structures for the design of efficient algorithms
CO 3	Understand Advanced Heap Structures suitable for solving Computational problems involving Optimisation and analysing these data structures using amortised analysis.
CO 4	Understand Advanced Graph algorithms suitable for solving advanced computational problems
CO 5	Understand the basic operation of Blockchaining along with the data structures used in it and the challenges in Blockchain data.

Mapping of course outcomes with program outcomes

	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11	PO 12
CO 1	3	2	2		1							
CO 2	2	2	3	2	1		1					
CO 3	2	3	3	2	1		1					
CO 4	3	3	2	1	2		1					
CO 5	3	2	2	2	3		1					

Assessment Pattern

Bloom's Category	Continuous Assessment Tests		End Semester Examination
	1	2	
Remember	10	10	10
Understand	20	20	20
Apply	20	20	30
Analyse			
Evaluate			
Create			

Mark distribution

Total Marks	CIE	ESE	ESE Duration
100	40	60	3 hours

Continuous Internal Evaluation Pattern:

Attendance	: 8 marks
Continuous Assessment Test (2 numbers)	: 20 marks
Assignment/Quiz/Course project	: 12 marks

End Semester Examination Pattern: There will be two parts; Part A and Part B. Part A contains 10 compulsory short answer questions, 2 from each module. Each question carries 3 marks. Part B contains 2 questions from each module of which student should answer any one. Each question can have maximum 2 sub-divisions and carry 6 marks.

Course Level Assessment Questions

Course Outcome 1 (CO1):

(a) Review the basic data structures such as array, linked list, stack, queue etc.
(b) Understand the set data structure and its implementation
(c) Understand the Disjoint set data structure
(d) Learn the basics of Amortised Analysis and its important types

Course Outcome 2 (CO2)

(a) Understand Balanced Binary Search Trees and the idea of Rotations
(b) Understand Red Blak Trees and their operations
(c) Understand B Trees and operations
(d) Obtain a basic awareness of Splay Trees and Suffix Trees.

Course Outcome 3(CO3):

(a) Understand the concepts of Mergeable Heaps and their operations.
(b) Understand the Binomial Heaps and its operations along with their amortised analysis
(c) Understand the Fibonacci Heaps and its operations along with their amortised analysis

Course Outcome 4 (CO4):

(a) Understand Graphs traversal techniques and topological sorting using these
(b) Understand the algorithms for finding the strongly connected components and biconnected components in a graph.
(c) Understand the Prim's and Krusksl's algorithms and their implementation
(d) Understand the Dijikstra's Single Source Shortest path algorithm and implementing it using Advanced Heap Structures.

Course Outcome 5 (CO5):

(a) Understand a basic overview of the Blockchain system architecture.
(b) Understand the Blockchain Data Structures and Data Types.
(c) Understand the problems and challenges in Blockchain data.

Model Question paper

Reg No.:_______________		Name:_____________________________________ _	
APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY FIRST SEMESTER M.C.A.DEGREE EXAMINATION, MODEL QUESTION PAPER			
Course Code: 20MCA105			
Course Name: ADVANCED DATA STRUCTURES			
Max. Marks: 60		Duration: 3 Hours	
PART A			

			Marks
		Answer all questions, each carries 3 marks.	Mar ks
1		What is meant by Hashing ?	(3)
2		How does Amortised Analysis differ from Average Case Analysis?	(3)
3		What is meant by Balanced Binary Search Tree? Give an example for a balanced binary search tree and an unbalanced one.	(3)
4		What is meant by Suffix Tree?	(3)
5		Give a valid Binomial heap with nodes 3,5,7,10,12,15.	(3)
6		Explain how fibonacci heaps are implemented?	(3)
7		What do you mean by Minimum Costs Spanning Tree?	(3)
8		What is meant by Strongly Connected Components? Illustrate with an example	(3)
9		What is meant by Block Chaining?	(3)
10		What is Contract Data ?	(3)
PART B			
Answer any one question from each module. Each question carries 6 marks.			
Module I			
11		How do you perform Amortised Analysis using Accounting method? Illustrate with Multipop Stack example.	(6)
OR			
12		Explain any three Hashing functions.	(6)
Module II			
13		What is meant by Red Black Tree? Explain how insertion is done in a Red Black Tree.	(6)
OR			
14		Give notes on B-Trees and Splay Trees.	(6)
Module III			
15		Explain how the Decrease-Key operation is performed on Binomial Heaps. What is the Amortised Cost of this operation?	(6)
OR			
16		Describe how the Delete-Key operation is performed in a Fibonacci heap? Illustrate with an example.	(6)
Module IV			
17		Explain the Breadth First Search algorithm with a suitable example.	(6)
OR			
18		Explain the Prim’s algorithm with an example.	(6)

		Module V	
19		Explain the Blockchain architecture in detail.	(6)
		OR	
20		Explain the problems to be solved in Blockchain Data Analysis.	(6)
		****	

Syllabus

Module 1 [12 hrs]

Review of basic data structures- Array, linked list and its variants, Stack ,Queue and Trees
Set Data Structure:- Representation of sets, Set implementation using bit string.
Hashing :- SImple hash functions, Collision and Collision Resolution techniques
Amortised Analysis - Aggregate, Accounting and Potential Methods (using the examples Multipop Stack and Incrementing Binary Counter only)
Disjoint sets- representations, Union, Find algorithms

Module 2 [10 hrs]

Advanced Tree Structures:- Balanced Binary Search trees, Red-Black trees- Properties of Red Black trees, Rotations, Insertion, Deletion. B-Trees- Basic operations on B-Trees – Insertion and Deletion, Introduction to Splay Trees and Suffix Trees

Module 3 [10 hrs]

Advanced Heap Structures:- Mergeable Heaps and operations on Mergeable Heaps. Binomial Heaps, Binomial Heap operations and Analysis, Fibonacci Heaps, Fibonacci Heap operations and Analysis**.**

Module 4 [14 hrs]

Advanced Graph Structures : Representation of graphs, Depth First and Breadth First Traversals, Topological Sorting, Strongly connected Components and Biconnected Components Minimum Cost Spanning Tree algorithms- Prim's Algorithm, Kruskal' Algorithm,. Shortest Path Finding algorithms – Dijikstra's single source shortest paths algorithm

Module 5[8 hrs]

Blockchain Data Structure:- Blockchain Architecture, Blockchain Data Structures and Data types, Contract Data, Problems to be solved in Blockchain data analysis

Text Books

1. Cormen T.H., Leiserson C.E, Rivest R.L. and Stein C, ***Introduction to Algorithms**,* Prentice Hall India, New Delhi, 2004 [Modules 1 to 4]

2. Yang, Xiaojing, Jinshan Liu, and Xiaohe Li. "***Research and Analysis of Blockchain Data." Journal of Physics: Conference Series. Vol. 1237***. No. 2. IOP Publishing, 2019.

Reference Books

1. Kleinberg, Jon, and Eva Tardos. ***Algorithm design***. Pearson Education India, 2006.

2. Aho A.V., Hopcroft J.E., and Ullman J.D., ***Data Structures and Algorithms***, Pearson Education, New Delhi, 1983.

3. Sahni S., ***Data Structures, Algorithms, and Applications in C++,*** Mc Graw Hill, Singapore, 1998.

Course Contents and Lecture Schedule

No	Topic	No. of Lecture Hours
1	**Review of basic data structures**	**10Hrs**
1.1	Array, Stack and Queue	
1.2	Linked list and its variants	
1.3	Representation of sets, Set implementation using bit string.	
1.4	Hashing – SImple hash functions	
1.5	Collision and Collision Resolution techniques	
1.6	Amortised Analysis	
1.7	Aggregate Method (Multipop Stack and Incrementing Binary Counter)	
1.8	Accounting Method (Multipop Stack and Incrementing Binary Counter)	
1.9	Potential Method (Multipop Stack and Incrementing Binary Counter)	
1.10	Disjoint sets- representations	
1.11	Union, Find algorithms	
2	**Advanced Tree Structures**	**10Hrs**
2.1	Balanced Binary Search trees	
2.2	Red-Black trees	
2.3	Properties of Red Black trees	
2.4	Rotations	
2.5	Insertion	

2.6	Deletion	
2.7	B-Trees	
2.8	Insertion and Deletion	
2.9	Splay Trees	
2.10	Suffix Trees	
3	**Advanced Heap Structures**	**8Hrs**
3.1	Mergeable Heaps	
3.2	Operations on Mergeable Heaps	
3.3	Binomial Heaps	
3.4	Binomial Heaps operations and Analysis	
3.5	Fibonacci Heaps	
3.6	Fibonacci Heap operations and Analysis.	
4	**Advanced Graph Structures**	**12Hrs**
4.1	Representation of graphs	
4.2	Depth First and Breadth First Traversals	
4.3	Topological Sorting	
4.4	Strongly connected Components	
4.5	Biconnected Components	
4.6	Minimum Cost Spanning Tree	
4.7	Prim's Algorithm	
4.8	Kruskal’s Algorithm	
4.9	Dijikstra’s single source shortest paths algorithm	
5	**Blockchain Data Structure**	**8Hrs**
5.1	Blockchain Architecture	
5.2	Blockchain Data Structures	
5.3	Blockchain Data types	
5.4	Contract Data	
5.5	Problems to be solved in Blockchain data analysis	

20MCA107	**ADVANCED SOFTWARE ENGINEERING**	**CATEGORY**	**L**	**T**	**P**	**CREDIT**
		GENERAL	**3**	**1**	**0**	**4**

Preamble:
Most of the programs on Computer Applications do not give due importance to teach Software Engineering in an Industry perspective. But this course, built upon the tools and techniques prevalent in Industry today, is supposed to make students Industry-ready.

Prerequisite: Programming proficiency in at least one of C, C++, Java, Python or PHP programming languages.

Course Outcomes: After the completion of the course the student will be able to

CO 1	Get a full view of the Software life cycle
CO 2	Gain a deep knowledge of Software Planning, Analysis and Design and Software Engineering Models
CO 3	Have a great comprehension of Coding Practices, Version Control using 'git' and Software Quality
CO 4	Acquire ample grasp of Design Patterns
CO 5	Get deeply familiarised with Software Testing and its automation
CO 6	Start using Agile Methodology
CO 7	Begin to apply CI/CD techniques in Software development

Mapping of course outcomes with program outcomes

	PO 1	**PO 2**	**PO 3**	**PO 4**	**PO 5**	**PO 6**	**PO 7**	**PO 8**	**PO 9**	**PO 10**	**PO 11**	**PO 12**
CO 1		2	2					3			1	1
CO 2		3	3					3				
CO 3					3				3	2	2	
CO 4			3		3							
CO 5					3					2	3	
CO 6					2			2	2		2	3
CO 7					3			1		2		

Assessment Pattern

Bloom's Category	Continuous Assessment Tests		End Semester Examination
	1	2	
Remember	10	10	10
Understand	20	20	20
Apply	10	10	20
Analyse			
Evaluate			
Create	10	10	10

Mark distribution

Total Marks	CIE	ESE	ESE Duration
100	40	60	3 hours

Continuous Internal Evaluation Pattern:

Attendance	: 8 marks
Continuous Assessment Test (2 numbers)	: 20 marks
Assignment/Quiz/Course project	: 12 marks

End Semester Examination Pattern: There will be two parts; Part A and Part B. Part A contains 10 compulsory short answer questions, 2 from each module. Each question carries 3 marks. Part B contains 2 questions from each module of which student should answer any one. Each question can have maximum 2 sub-divisions and carry 6 marks.

Course Level Assessment Questions

Course Outcome 1 (CO1):

(a) Understand the software development as an engineering process and its stages.

(b) Understand Software development lifecycle (SDLC).

(c) Understand software engineering models.

(d) Learn how to prepare software requirements specification, approaches and methodologies to prepare requirement specifications document.

Course Outcome 2 (CO2)

(a) Understand writing industry-grade software programs, following style guides and coding standards.

(b) Learn core concepts of software version control system and common operations with Git distributed version control system.

(c) Understanding software quality concepts with respect to software requirement specifications document, what to conform to at various stages of SDLC.

(d) Understand what to ensure at various stage of SDLC to ensure quality of developed software system.

Course Outcome 3(CO3):

(a) Learn Object Oriented Programming concepts comprehensively.

(b) Learn the concept of Design Patterns, category of patterns, and how to select appropriate design patterns.

(c) Understand Unit testing concepts and xUnit architecture.

(d) Learn Unit testing frameworks and writing unit testing for Java and one of PHP or Python.

(e) Understand the concepts Continuous Integration and Continuous Delivery (CICD).

Course Outcome 4 (CO4):

(a) Knowledge of Git distributed version control system to use in a product environment.

(b) Knowledge of OOP paradigm and software Design Patterns to design the software system.

(c) Knowledge of unit testing frameworks such as Junit, uniitest, phpdbg for wiring units tests in a software production environment.

(d) Knowledge of software testing CI/CD practices.

Course Outcome 5 (CO5):

(a) Understand software testing concepts and principles.

(b) Learn common approaches to ensure software quality through testing.

(c) In-depth understanding of various types of testing methodologies.

(d) Learn about testing automation and understand commonly used test automation types.

(e) Learn to use Robot framework.

Course Outcome 6 (CO5):

(a) Understand the concepts of Agile methodology.

(b) Learn to use Scrum framework for implementing Agile methodology for executing a software development process.

(c) Learn to monitor a software development project using a Scrum tool.

Course Outcome 7 (CO5):

(a) Understand the concepts of Software Configuration Management.

(b) Learn about build and deployment environments.

(c) Understand the concepts of Continuous Integration and essential practices.

(d) Understand the concepts of deployment automation and learn to use Ansible.

Model Question paper

APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY
First Semester MCA Degree Examination (R&S)

Course Code: 20MCA107
Course Name: ADVANCED SOFTWARE ENGINEERING

Total Marks: 60 Duration: 3 Hours

PART A

Answer all questions, each carries3 marks. Marks

1. Why is Software Engineering important? (3)
2. What are the desired requirements of a good software engineering model? (3)
3. What is the purpose of a version control system? (3)
4. Explain the different ways to fix commits in Git (3)
5. What is anti-pattern? (3)
6. What is an abstract test? (3)
7. Distinguish between black box testing and white box testing. (3)
8. Draw a model Sprint Backlog for the login module of a simple web portal (3)

9. Write a short note on release candidate (3)

10. Differentiate continuous delivery and continuous deployment (3)

PART B

Required to answer one question from each module in full.
Each module carries 6 marks for either of the questions.

Module I

11. Prepare a basic Software Requirements Specification for Savings Bank accounts. (6)

OR

12. How is Use Case different from User Stories? Enlist the advantage of each. (6)

Module II

13. How do you create, switch and view branches in Git? explain how to merge commits between branches. (6)

OR

14. You have cloned a repository which was then modified by another developer. You make changes locally and try to execute push. What are the possible outputs? How will you solve the problems, if any? (6)

Module III

15. Explain the important design patterns. (6)

OR

16. When are assertions and expected error tests used in Unit tests? (6)

Module IV

17. Write down the scrum. (6)

OR

18. Differentiate Black box testing and White box testing. Give appropriate example for each for "only black box testing is possible" and "necessary to do white box testing" scenarios. (6)

Module V

19. Explain the strategy for implementing Continuous integration. (6)

OR

20. What is a deployment pipeline? Explain the anatomy of a deployment pipeline with a (6) neat diagram. Comment on the various stages of a deployment pipeline.

Syllabus

Module 1 [8 hrs]

Introduction to Software Engineering: What is Software Engineering, Characteristics of Software.

Life cycle of a software system: software design, development, testing, deployment, Maintenance.

Project planning phase: project objectives, scope of the software system, empirical estimation models, COCOMO, staffing and personnel planning.

Software Engineering models: Predictive software engineering models, model approaches, prerequisites, predictive and adaptive waterfall, waterfall with feedback (Sashimi), incremental waterfall, V model; Prototyping and prototyping models.

Software requirements specification, Eliciting Software requirements, Requirement specifications, Software requirements engineering concepts, Requirements modelling, Requirements documentation. Use cases and User stories.

Module 2 [10 hrs]

Programming Style Guides and Coding Standards; Literate programming and Software documentation; Documentation generators, Javadoc, phpDocumentor.

Version control systems basic concepts; Concept of Distributed version control system and Git; Setting up Git; Core operations in Git version control system using command line interface (CLI): Clone a repository; View history; Modifying files; Branching; Push changes, Clone operation, add, commit, log, diff commands, conflict resolution. Pushing changes to the master; Using Git in IDEs and UI based tools.

Software Quality: Understanding and ensuring requirements specification quality, design quality, quality in software development, conformance quality.

Module 3 [10 hrs]

OOP Concepts; Design Patterns: Basic concepts of Design patterns, How to select a design pattern, Creational patterns, Structural patterns, Behavioural patterns. Concept of Anti-patterns.

Unit testing and Unit Testing frameworks, The xUnit Architecture, Writing Unit Tests using at least one of Junit (for Java), unittest (for Python) or phpdbg (PHP). Writing tests with Assertions, defining and using Custom Assertions, single condition tests, testing for expected errors, Abstract test.

Module 4 [10 hrs]

Concepts of Agile Development methodology; Scrum Framework.

Software testing principles, Program inspections, Program walkthroughs, Program reviews; Blackbox testing: Equivalence class testing, Boundary value testing, Decision table testing, Pairwise testing, State transition testing, Use-case testing; White box testing: control flow testing, Data flow testing.

Testing automation: Defect life cycle; Regression testing, Testing automation; Testing non-functional requirements.

Module 5[10 hrs]

Software Configuration Management: Using version control, Managing dependencies, Managing software configuration, Managing build and deployment environments.

Continuous Integration: Prerequisites for continuous integration, Essential practices.

Continuous Delivery: Principles of Software delivery, Introduction and concepts.

Build and deployment automation, Learn to use Ansible for configuration management.

Test automation (as part of continuous integration), Learn to set up test automation cases using Robot Framework.

Notes

1. At the end of Module 1, conduct the following class work with appropriate evaluation points: Prepare Software Specification Document for a moderately complex process flow system (*e.g.* Broadband fault booking and resolution system covering technical, operational and commercial aspects, covering organizational and subscriber use cases).

2. At the end of Module 2, clone an open source project using Git and perform all based operations.

Reference Books

1. Philip A. Laplante, *What Every Engineer Should Know about Software Engineering*, CRC Press [Module 1]
2. Murali Chemuturi, *Mastering Software Quality Assurance: Best Practices, Tools and Technique for Software Developers*, J Ross Publishing [Module 2]
3. Ben Straub, Scott Chacon, *Pro Git*, 2nd Edition, Apress [Module 2]
4. Erich Gamma et. al., *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison-Wesley [Moule 3]
5. Vaskaran Sarcar, *Java Design Patterns: A Hands-On Experience with Real-World Examples*, Apress [Module 3]
6. Alistair Cockburn and Robert Cecil Martin, *Agile Software Development: The Cooperative Game (2nd edition)*, Addition Wesley [Module 4]

7. Ken Schwaber , *Agile Software Development with Scrum*, Pearson [Module 4]
8. Lisa Crispin, Agile Testing: *A Practical Guide for Testers and Agile Teams*, Adison Wesley
9. Paul Hamill, *Unit Test Frameworks*, O'Reilly Media [Module 4]
10. Glenford J. Myers, et. al., *The Art of Software Testing*, Wiley [Module 4, 5]
11. Lee Copeland, *A Practitioner's Guide to Software Test Design*, Artech House Publishers [Module 4, 5]
12. Jez Humble and David Farley, *Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation,* Pearson Education [Module 5]

Web-based Resources

1. *Git Handbook* https://guides.github.com/introduction/git-handbook/ Retrieved 8 July 2020 [Module 2]
2. *Git User Manual* https://mirrors.edge.kernel.org/pub/software/scm/git/docs/user-manual.html Retrieved 8 July 2020 [Module 2]
3. *Introduction to Software Engineering/Quality* https://en.wikibooks.org/wiki/Introduction_to_Software_Engineering/Quality Retrieved 8 July 2020 [Module 2]
4. *Understanding software design patterns* https://opensource.com/article/19/7/understanding-software-design-patterns Retrieved 8 July 2020 [Module 3]
5. *The Scrum Guide* https://www.scrumguides.org/docs/scrumguide/v2017/2017-Scrum-Guide-US.pdf Retrieved 8 July 2020 [Module 4]
6. *unittest — Unit testing framework* https://docs.python.org/3/library/unittest.html Retrieved 8 July 2020 [Module 4]
7. What is CI/CD? https://www.redhat.com/en/topics/devops/what-is-ci-cd Retrieved 8 July 2020 [Module 5]

Course Contents and Lecture Schedule

No	Topic	No. of Lecture Hours
1	**Software Engineering**	
1.1	What is Software Engineering, Characteristics of Software Engineering	1
1.2	Life cycle of a software system	1
1.3	Project planning	1
1.4	Software Engineering Models	2
1.5	Software Requirements Specification	3
2	**Industry Best Practices**	
2.1	Programming style guides and coding standards	1
2.2	Software version control systems, basic concepts	1
2.3	Git distributed version control system, introduction	2
2.4	Common operations in Git	4

No	Topic	No. of Lecture Hours
2.5	Software quality, achieving	2
3	**System Design Methodologies**	
3.1	Object Oriented Programming	1
3.2	Software Design Patterns	4
3.3	Unit Testing concepts and xUnit architecture	1
3.4	Unit testing frameworks: Junit, unittest, phpdbg	2
3.5	Writing unit test code	2
4	**Agile Development Methodology**	
4.1	Agile Development methodology, introduction	2
4.2	Scrum framework	5
4.3	Automated testing	3
5	**Continuous Integration and Continuous Development (CI/CD)**	
5.1	Configuration Management	2
5.2	Continuous Integration, concepts and practices	2
5.3	Continuous Delivery, concepts and practices	2
5.4	Build and deployment automation	2
5.5	Test automation for CI/CD	2

20MCA131	PROGRAMMING LAB	CATEGORY	L	T	P	CREDIT
		PRACTICAL	0	1	3	2

Preamble: This course introduces a basic step towards program writing and develops the logical ability and problem-solving skill using Python Programming Language. Students are able to do testing and debugging of code written in Python.

Prerequisite: None

Course Outcomes: After the completion of the course the student will be able to

CO 1	Understands basics of Python Programming language including input/output functions, operators, basic and collection data types
CO 2	Implement decision making, looping constructs and functions
CO 3	Design modules and packages - built in and user defined packages
CO 4	Implement object-oriented programming and exception handling.
CO 5	Create files and form regular expressions for effective search operations on strings and files.

Mapping of course outcomes with program outcomes

	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11	PO 12
CO 1	2	2	2	1	2							
CO 2	3	3	3	2	2							
CO 3	3	3	3	3	3						1	
CO 4	3	3	3	3	3						1	
CO 5	3	3	3	3	3						1	

Assessment Pattern

Bloom's Category	Continuous Assessment Tests		End Semester Examination
	1	2	
Remember (K1)			
Understand (K2)			
Apply (K3)	10	10	10
Analyse (K4)	10	10	10
Evaluate (K5)	10	10	10
Create(K6)	20	20	20

Mark distribution

Total Marks	CIE	ESE	ESE Duration
100	50	50	3 hours

Continuous Internal Evaluation Pattern:

Maximum Marks: 50	
Attendance	15%
Maintenance of daily lab record and GitHub management	20%
Regular class viva	15%
Timely completion of day to day tasks	20%
Tests/Evaluation	30%

End Semester Examination Pattern:

Maximum Marks: 50			
Verification of Daily program record and Git Repository			5 marks
Viva			10 marks
Problem solving (Based on difficulty level, one or more questions may be given)	Flowchart / Algorithm / Structured description of problem to explain how the problem can be solved / Interface Design	15%	35 marks
	Program correctness	50%	
	Code efficiency	15%	
	Formatted output and Pushing to remote Git repository	20%	
Total Marks			50 marks

Course Level Assessment Questions

Course Outcome 1 (CO1):

1. Familiarizing Text Editor, IDE, Code Analysis Tools etc // Use any IDE like PyCharm, PyDev…
2. Display future leap years from current year to a final year entered by user.
3. List comprehensions:
   (a) Generate positive list of numbers from a given list of integers
   (b) Square of N numbers
   (c) Form a list of vowels selected from a given word
   (d) List ordinal value of each element of a word (Hint: use ord() to get ordinal values)
4. Count the occurrences of each word in a line of text.
5. Prompt the user for a list of integers. For all values greater than 100, store 'over' instead.
6. Store a list of first names. Count the occurrences of 'a' within the list
7. Enter 2 lists of integers. Check (a) Whether list are of same length (b) whether list sums to same value (c) whether any value occur in both
8. Get a string from an input string where all occurrences of first character replaced with '$', except first character.
   [eg: onion -> oni$n]
9. Create a string from given string where first and last characters exchanged. [eg: python -> nythop]
10. Accept the radius from user and find area of circle.
11. Find biggest of 3 numbers entered.
12. Accept a file name from user and print extension of that.
13. Create a list of colors from comma-separated color names entered by user. Display first and last colors.
14. Accept an integer n and compute n+nn+nnn.
15. Print out all colors from color-list1 not contained in color-list2.
16. Create a single string separated with space from two strings by swapping the character at position 1.
17. Sort dictionary in ascending and descending order.

18. Merge two dictionaries.
19. Find gcd of 2 numbers.
20. From a list of integers, create a list removing even numbers.

Course Outcome 2 (CO2)

1. Program to find the factorial of a number
2. Generate Fibonacci series of N terms
3. Find the sum of all items in a list
4. Generate a list of four digit numbers in a given range with all their digits even and the number is a perfect square.
5. Display the given pyramid with step number accepted from user.

```
Eg: N=4
1
2  4
3  6  9
4  8 12  16
```

6. Count the number of characters (character frequency) in a string.
7. Add ‘ing’ at the end of a given string. If it already ends with ‘ing’, then add ‘ly’
8. Accept a list of words and return length of longest word.
9. Construct following pattern using nested loop

```
*
* *
* * *
* * * *
* * * * *
* * * *
* * *
* *
*
```

10. Generate all factors of a number.

11. Write lambda functions to find area of square, rectangle and triangle.

Course Outcome 3(CO3):

1. Work with built-in packages
2. Create a package graphics with modules rectangle, circle and sub-package 3D-graphics with modules cuboid and sphere. Include methods to find area and perimeter of respective figures in each module. Write programs that finds area and perimeter of figures by different importing statements. (Include selective import of modules and import * statements)

Course Outcome 4 (CO4):

1. Create Rectangle class with attributes length and breadth and methods to find area and perimeter. Compare two Rectangle objects by their area.
2. Create a Bank account with members account number, name, type of account and balance. Write constructor and methods to deposit at the bank and withdraw an amount from the bank.
3. Create a class Rectangle with private attributes length and width. Overload '<' operator to compare the area of 2 rectangles.
4. Create a class Time with private attributes hour, minute and second. Overload '+' operator to find sum of 2 time.

5. Create a class Publisher (name). Derive class Book from Publisher with attributes title and author. Derive class Python from Book with attributes price and no_of_pages. Write a program that displays information about a Python book. Use base class constructor invocation and method overriding.

Course Outcome 5 (CO5):

1. Write a Python program to read a file line by line and store it into a list.
2. Python program to copy odd lines of one file to other
3. Write a Python program to read each row from a given csv file and print a list of strings.
4. Write a Python program to read specific columns of a given CSV file and print the content of the columns.
5. Write a Python program to write a Python dictionary to a csv file. After writing the CSV file read the CSV file and display the content.

Syllabus:

Input, Output and Import Functions, Operators, Data Types, Decision Making & Loops, Functions, Modules and Packages, File Handling, Object Handling, Exception Handling, Regular Expressions

Reference Books

1. Wesley J. Chun, “***Core Python Applications Programming***”, 3rd Edition , Pearson Education, 2016
2. Charles Dierbach, “***Introduction to Computer Science using Python***”, Wiley, 2015
3. Jeeva Jose, “***Taming Python by Programming***”, Khanna Publishers, New Delhi, 2018
4. Downey, A. et al., "***How to think like a Computer Scientist: Learning with Python***", John Wiley, 2015

Web References

1. https://archive.org/details/MIT6.00SCS11
2. https://www.coursera.org/course/pythonlearn
3. http://www.learnerstv.com/Free-Computer-Science-Video-lectures-ltv163-Page1.htm
4. https://www.coursera.org/learn/python-databases

Course Contents and Lab Schedule

Topic	**No. of hours**
1. Input, Output and Import Functions	3
2. Operators	5
3. Data Types	6
4. Decision Making & Loops	6
5. Functions	5
6. Modules and Packages	6
7. File Handling	5
8. Object Handling	5
9. Exception Handling	2
10. Regular Expressions	4

20MCA133	WEB PROGRAMMING LAB	CATEGORY	L	T	P	CREDIT
		PRACTICAL	0	1	3	2

Preamble: With a dynamic learn-by-doing focus, this laboratory course encourages the students to explore the designing of web application by implementing the relevant and recent techniques. This course challenges the students to exercise their creativity in both programming and designing.

Prerequisite: Basic understanding of computer programming, Internet and Database etc. is very helpful.

Course Outcomes: After the completion of the course the student will be able to;

CO 1	Explore markup languages features and create interactive web pages using them.
CO 2	Learn and design client-side validation using scripting languages.
CO 3	Design front end web page and connect to the back-end databases.
CO 4	Do Client-side & Server-side scripting
CO 5	Develop Web Applications

Mapping of course outcomes with program outcomes

	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11	PO 12
CO 1	3	3	3	2	2		3	3				
CO 2	3	3	3	2	2		3	3	1			
CO 3	3	3	3	2	2		3	3				
CO 4	3	3	3	2	2		3	3				2
CO 5	3	3	3	3	3		3	3			2	2

Assessment Pattern

Bloom's Category	**Continuous Assessment Tests**		**End Semester Examination**
	1	**2**	
Remember(K1)			
Understand(K2)			
Apply(K3)	10	10	10
Analyse(K4)	10	10	10

Evaluate(K5)	10	10	10
Create(K6)	20	20	20

Mark distribution

Total Marks	CIE	ESE	ESE Duration
100	50	50	3 hours

Continuous Internal Evaluation Pattern:

Maximum Marks: 50	
Attendance	15%
Maintenance of daily lab record and GitHub management	20%
Regular class viva	15%
Timely completion of day to day tasks	20%
Tests/Evaluation	30%

End Semester Examination Pattern:

Maximum Marks: 50			
Verification of Daily program record and Git Repository			5 marks
Viva			10 marks
Problem solving (Based on difficulty level, one or more questions may be given)	Flowchart / Algorithm / Structured description of problem to explain how the problem can be solved / Interface Design	15%	35 marks
	Program correctness	50%	
	Code efficiency	15%	
	Formatted output and Pushing to remote Git repository	20%	
Total Marks			50 marks

Course Level Assessment Questions

Course Outcome 1 (CO1):

1. Model a simple HTML file to demonstrate the use of different tags. (K3)
2. Create a HTML file to link to different HTML page which contains images, tables, and also link within a page. (K6)
3. Create a HTML page with different types of frames such as floating frame, navigation frame & mixed frame. (K6)
4. Analyze CSS by applying the different styles using inline, external & internal style sheets in a HTML file. (K4)
5. Demonstrate a registration form using HTML. (K3)

Course Outcome 2 (CO2)

1. Create a HTML page to explain the use of various predefined functions in a string and math object in java script. (K6)
2. Generate the calendar using JavaScript code by getting the year from the user. (K6)
3. Create a HTML registration form and to validate the form using JavaScript code. (K6)
4. Evaluating JavaScript Event Handling for every click of a button to change the background color of a HTML page. (K5)
5. Create a HTML page to display a new image and text when the mouse comes over the existing content in the page using JavaScript Event Handling. (K6)
6. Create a HTML page to show online exam using JavaScript. (K6)

Course Outcome 3(CO3):

1. Develop a PHP program to connect to a database and retrieve data from a table and show the details in a neat format. (K6)

Course Outcome 4 (CO4):

1. Outline a registration form using PHP and do necessary validations. (K4)
2. Compose Electricity bill from user input based on a given tariff using PHP. (K6)

3. Build a PHP code to store name of students in an array and display it using print_r function. Sort and Display the same using asort & arsort functions. (K6)
4. Build a PHP code to store name of Indian Cricket players in an array and display the same in HTML table. (K6)

Course Outcome 5 (CO5):

1. Develop Web applications using HTML and PHP and deploy. (K6)
2. Using PHP and MySQL, develop a program to accept book information viz. Accession number, title, authors, edition and publisher from a web page and store the information in a database and to search for a book with the title specified by the user and to display the search results with proper headings. (K6)
3. Develop a web application for Airline Reservation System using any PHP framework (Laravel, CodeIgniter, Symfony, CakePHP etc.). (K6)
4. Test the application on an Application Server. (K5)

Syllabus

Introduction To Web: Client/Server concepts, Components of Web Application, Types of Web Content, Overview of HTTP - HTTP request – response, Generation of dynamic web pages, Application Servers, Web Security.

Markup Language (HTML): Formatting and Fonts, Commenting Code, Anchors, Backgrounds, Images, Hyperlinks, Lists, Tables, Frames, HTML Forms.

Cascading Style Sheet (CSS): The need for CSS, Basic syntax and structure, Inline Styles, Embedding Style Sheets, Linking External Style Sheets, Backgrounds, Manipulating text, Margins and Padding, Positioning using CSS.

Client Side Scripting using JavaScript: Core features, Data types and Variables, Operators -Expressions and Statements, Functions, Objects, Array, String - Date and Math related Objects, Document Object Model, Event Handling, Form handling and validations.

An overview of Relational Database Design: Tables, Attributes, Tuples, Primary keys, Foreign keys, Indexes, DDL Commands – CREATE, ALTER, DROP and TRUNCATE; DML Commands – SELECT, INSERT, UPDATE and DELETE.

Server Side Scripting using PHP: Setting up the environment (Example - XAMP server), PHP Programming basics - Print/echo, Variables and constants, Strings and Arrays, Operators, Control structures and looping structures, Functions, Reading Data in Web Pages, Embedding PHP within HTML, Establishing connectivity with database, Debugging with phpdbg.

Web Application development in any PHP framework (Laravel, CodeIgniter, Symfony, CakePHP etc.): Naming convention, MVC model, Connectivity with Database, Database interaction.

Debugging web apps: Browser debugging tools (Any browser web developer tools) - View and change the DOM and CSS, Console, Debug JavaScript, View and debug network activity, Performance tools etc.

Reference Books

1. David Flanagan, "***JavaScript: The Definitive Guide***", 6th Edition", O'Reilly Media
2. Douglas E Comer, "***The Internet Book: Everything You Need to Know About Computer Networking and How the Internet Works***", 4th Edition, Prentice Hall
3. Harvey Deitel and Abbey Deitel, "***Internet and World Wide Web - How To Program***", 5th Edition,

   Pearson Education
4. Abraham Silberschatz, Henry F. Korth, S. Sudarshan, "***Database System Concepts***", McGraw Hill Education, 6th Edition (2011)
5. Steve Suehring, Tim Converse, and Joyce Park, "***PHP6 and MySQL Bible***", Wiley India Pvt Ltd (2009)
6. Steven Holzner, "***PHP-The Complete Reference***", Tata McGraw Hill, 1st Edition (2007)
7. Thomas A Powell, Fritz Schneider, "***JavaScript: The Complete Reference***", 3rd Edition, Tata McGraw Hill

Web Resources

1. http://php.net/manual/
2. https://pepa.holla.cz/wp-content/uploads/2016/08/JavaScript-The-Definitive-Guide-6th-Edition.pdf
3. http://index-of.es/PHP/PHP6%20and%20MySQL%20Bible.pdf
4. https://www.udemy.com/course/html5-fundamentals-for-beginners/
5. https://www.udemy.com/course/programming-in-javascript/
6. https://www.udemy.com/course/php-mysql-tutorial/

List of Lab Experiments/Exercises

1. Create a simple HTML file to demonstrate the use of different tags.
2. Create a HTML file to link to different HTML page which contains images, tables, and also link within a page.
3. Create a HTML page with different types of frames such as floating frame, navigation frame & mixed frame.
4. Create a HTML file by applying the different styles using inline, external & internal style sheets.
5. Create a registration form using HTML.
6. Create a HTML page to explain the use of various predefined functions in a string and math object in java script.
7. Generate the calendar using JavaScript code by getting the year from the user.
8. Create a HTML registration form and to validate the form using JavaScript code.
9. Create a HTML page to change the background color for every click of a button using JavaScript Event Handling.
10. Create a HTML page to display a new image and text when the mouse comes over the existing content in the page using JavaScript Event Handling.
11. Create a HTML page to show online exam using JavaScript.
12. Develop a registration form using PHP and do necessary validations.
13. Compose Electricity bill from user input based on a given tariff using PHP.
14. Build a PHP code to store name of students in an array and display it using print_r function. Sort and Display the same using asort & arsort functions.
15. Build a PHP code to store name of Indian Cricket players in an array and display the same in HTML table.
16. Develop a PHP program to connect to a database and retrieve data from a table and show the details in a neat format.
17. Develop Web applications using HTML and PHP and deploy.
18. Using PHP and MySQL, develop a program to accept book information viz. Accession number, title, authors, edition and publisher from a web page and store the information in a database and to search for a book with the title specified by the user and to display the search results with proper headings.

19. Develop a web application for Airline Reservation System using any PHP framework (Laravel, CodeIgniter, Symfony, CakePHP etc.).
20. Test the application on an Application Server.

Note: *Students can be given a group micro project, so that they learn to work in a team environment. They can also be trained on project management tools.*

Course Contents and Lecture Schedule

Topic	**No. of lectures**
Client/Server concepts, Components of Web Application, Types of Web Content, Overview of HTTP - HTTP request – response, Generation of dynamic web pages, Application Servers, Web Security.	1Hr.
HTML - Formatting and Fonts, Commenting Code, Anchors,Backgrounds, Images, Hyperlinks.	4 Hrs.
HTML - Lists, Tables, Frames, HTML Forms.	4 Hrs.
The need for CSS, Basic syntax and structure, Inline Styles, Embedding Style Sheets, Linking External Style Sheets, Backgrounds.	4 Hrs.
CSS - Manipulating text, Margins and Padding, Positioning using CSS.	4 Hrs.
JavaScript: Core features, Data types and Variables, Operators - Expressions and Statements.	3 Hrs.
JavaScript: Functions, Objects, Array, String - Date and Math related Objects, Document Object Model, Event Handling.	4 Hrs.
JavaScript: Form handling and validations.	4 Hrs.
An overview of Relational Database Design: Tables, Attributes, Tuples, Primary keys, Foreign keys, Indexes, DDL Commands – CREATE, ALTER, DROP and TRUNCATE.	4 Hrs.
DML Commands – SELECT, INSERT, UPDATE and DELETE.	4 Hrs.
PHP: Setting up the environment (Example - XAMP server), PHP Programming basics - Print/echo, Variables and constants.	4 Hrs.
Strings and Arrays, Operators, Control structures and looping structures.	4 Hrs.
Functions, Reading Data in Web Pages, Embedding PHP within HTML, Establishing connectivity with database.	4 Hrs.
PHP framework: naming convention, MVC model, Connectivity with Database, Database Interaction.	6 Hrs.

20MCA135	DATA STRUCTURES LAB	CATEGORY	L	T	P	CREDIT
		PRACTICAL	0	1	3	2

Preamble: This is the companion course of 20MCA105 Advanced Data Structures and provides the students hands-on experience of the advanced data structures which will boost up the knowledge and confidence of students in applying these techniques while dealing with real life computing problems.

Prerequisite: Basic Data Structures, Knowledge of any programming language, preferably 'C'.

Course Outcomes: After the completion of the course the student will be able to

CO 1	Use Debuggers, Profilers and advanced Compiler options.
CO 2	Implement the Set and Disjoint Set Data Structures.
CO 3	Understand the practical aspects of Advanced Tree Structures.
CO 4	Realise Modern Heap Structures for effectively solving advanced Computational problems.
CO 5	Implement Advanced Graph algorithms suitable for solving advanced computational problems.

Mapping of course outcomes with program outcomes

	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11	PO 12
CO 1	2				3		1					
CO 2	**3**	2	2		1							
CO 3	**2**	2	3	2	1		1					
CO 4	**2**	3	3	2	1		1					
CO 5	**3**	3	2	1	2		1					

Assessment Pattern

Bloom's Category	**Continuous Assessment Tests**		**End Semester Examination**
	1	**2**	
Remember(K1)			
Understand(K2)			
Apply(K3)	10	10	10
Analyse(K4)	10	10	10
Evaluate(K5)	10	10	10
Create(K6)	20	20	20

Mark distribution

Total Marks	**CIE**	**ESE**	**ESE Duration**
100	50	50	3 hours

Continuous Internal Evaluation Pattern:

Maximum Marks: 50	
Attendance	15%
Maintenance of daily lab record and GitHub management	20%
Regular class viva	15%
Timely completion of day to day tasks	20%
Tests/Evaluation	30%

End Semester Examination Pattern:

Maximum Marks: 50			
Verification of Daily program record and Git Repository			5 marks
Viva			10 marks
Problem solving (Based on difficulty level, one or more questions may be given)	Flowchart / Algorithm / Structured description of problem to explain how the problem can be solved / Interface Design	15%	35 marks
	Program correctness	50%	
	Code efficiency	15%	
	Formatted output and Pushing to remote Git repository	20%	
Total Marks			50 marks

Course Level Assessment Questions

Course Outcome 1 (CO1):

1. Write a C program ‘sum.c’ to add two numbers. Read the input from Standard Input and write output to Standard output. Compile and generate sum.out which is then debug with gdb.

2. Modify ‘sum.c’ by adding a function for finding the sum of two numbers. Then profile the executable with gprof.

Course Outcome 2 (CO2)

1. Create the Set ADT with Add, Remove, Union, Intersection and Difference operations. Implement using Bit Strings.

2. Implement the Disjoint set ADT with Create, Union and Find operations.

3. Implement Kruskal’s algorithm using Disjoint sets.

Course Outcome 3(CO3):

1. Implement B-Tree and its operations..
2. Implement Red Black Tree and the associated operations.

Course Outcome 4 (CO4):

1. Create the Binomial Heap ADT and implement the basic operations.
2. Use any Mergeable Heap to implement Single source shortest path algorithm.

Course Outcome 5 (CO5):

1. Finding the strongly connected components of a directed graph.
2. Prim's Algorithm for Minimum cost spanning tree.

Syllabus:

Based on the syllabus of 20MCA105 Advanced Data Structures.

Text Books

1. Cormen T.H., Leiserson C.E, Rivest R.L. and Stein C, ***Introduction to Algorithms***, Prentice Hall India, New Delhi, 2004

Reference Books

1. Kleinberg, Jon, and Eva Tardos. ***Algorithm design***. Pearson Education India, 2006.
2. Aho A.V., Hopcroft J.E., and Ullman J.D., ***Data Structures and Algorithms***, Pearson Education, New Delhi, 1983.
3. Sahni S., Data Structures, ***Algorithms, and Applications in C++***, Mc Graw Hill, Singapore, 1998.

Web Reference

1. **https://gcc.gnu.org/onlinedocs/gcc/Option-Summary.html**
2. **https://www.gnu.org/software/gdb/documentation/**
3. **https://ftp.gnu.org/old-gnu/Manuals/gprof-2.9.1/html_mono/gprof.html**

Course Contents and Lecture Schedule

Topic	**No. of hours**
1. Advanced use of gcc : Important Options -o, -c, -D, -l, -I, -g, -O, -save-temps, -pg	1
2. Familiarisation with gdb : Important Commands - break, run, next, print, display, help	1
3. Using gprof : Compile, Execute and Profile	1
4. Review of Basic Data Structures (Array, List, Stack, Queue,Trees) a. Merge two sorted arrays and store in a third array b. Circular Queue - Add, Delete, Search c. Singly Linked Stack - Push, Pop, Linear Search d. Doubly linked list - Insertion, Deletion, Search e. Binary Search Trees- Insertion, Deletion, Search	8
5. Set Data Structure and set operations (Union, Intersection and Difference) using Bit String.	3
6. Disjoint Sets and the associated operations (create, union, find)	3
7. Binomial Heaps and operations (Create, Insert, Delete, Extract-min, Decrease key)	4
8. B Trees and its operations	4
9. Red Black Trees and its operations	4
10. Graph Traversal techniques (DFS and BFS) and Topological Sorting	4
11. Finding the Strongly connected Components in a directed graph	3
12. Prim's Algorithm for finding the minimum cost spanning tree	3
13. Kruskal's algorithm using the Disjoint set data structure	3
14. Single Source shortest path algorithm using any heap structure that supports mergeable heap operations	3

SEMESTER 2