Jharkhand University of Technology

Ranchi, 834010

Diploma – Computer Science Engineering

SYLLABUS

For Diploma Program in

Computer Science Engineering

(Effective from 2024-25)

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING

(4th – SEMESTER)

Data Structures with Python

Subject code – CSE401

1. Rationale

Data structures are the techniques organizing data and of designing the algorithms for real-life projects. Knowledge of data structures is essential for software design and development. Learning data structures with Python offer flexibility and ease of programming with many built in data structures and libraries.

2. Course Outcomes:
At the end of the Course, the student will be able to:

CO-01	Explain data structures types, list their applications.
CO-02	Apply the right Algorithm design strategies to solve a given problem.
CO-03	Choose the right data structure to develop solution to a given computing problem.
CO-04	Analyse space and time complexities of the algorithm used and plot a graph.

3. Course Content

Week	CO	PO	Lecture (Knowledge Criteria)	Tutorial (Activity Criteria)	Practice (Performance Criteria)
1	01	1, 2, 3	Introduction to Data Structures, operations, classification, Characteristics. Primitive types – primitive data structures, python examples. Non primitive types - Non primitive data structures, python examples. Linear and nonlinear data structures – with python examples. Introduction, Abstractions, Abstract Data Types, An Example of Abstract Data Type (Student, Date, Employee), Defining the ADT, Using the ADT, Implementing the ADT.	**Refer Table 1**	1. Python program to Use and demonstrate basic data structures. 2. Implement an ADT with all its operations.
2	01, 02, 04	1, 2, 3, 4, 7	Algorithm Analysis – Space Complexity, Time Complexity. Run time analysis. Asymptomatic notations, Big-O Notation, Omega Notation, Theta Notation.		1. Implement an ADT and Compute space and time complexities. 2. Implement above solution using array and Compute space and time complexities and compare two solutions.

3	01, 02, 04	1, 2, 3, 4, 7	Algorithm design strategies: Brute force – Bubble sort, Selection Sort, Linear Search. Decrease and conquer - Insertion Sort.		1. Implement Linear Search compute space and time complexities, plot graph using asymptomatic notations. 2. Implement Bubble, Selection, insertion sorting algorithms compute space and time complexities, plot graph using asymptomatic notations.
4	01, 02, 04	1, 2, 3, 4, 7	Divide and conquer - Merge Sort, Quick Sort, Binary search. Dynamic programming - Fibonacci sequence Backtracking – Concepts only (Implementation examples with recursion inweek 9). Greedy – Concepts only.		1. Implement Binary Search using recursion Compute space and time complexities, plot graph using asymptomatic notations and compare two. 2. Implement Merge and quick sorting algorithms compute space and time complexities, plot graph using asymptomatic notations and compare all solutions. 3. Implement Fibonacci sequence with dynamic programing.
5	01, 02, 03, 04	1, 2, 3, 4,	Linear(arrays) vs nonlinear (pointer) structures – Run time and space requirements, when to use what? Introduction to linked list, Examples: Image viewer, music player list etc. (to be used to explain concept of list), applications.		1. Implement Singly linked list (Traversingthe Nodes, searching for a Node, Prepending Nodes, Removing Nodes)
6	01, 02, 03, 04	1, 2, 3, 4,	The Singly Linked List- Creating Nodes, Traversing the Nodes, searching for a Node, Prepending Nodes, Removing Nodes. Linked List Iterators.		1. Implement linked list Iterators.
7	01, 02, 03, 04	1, 2, 3, 4,	The Doubly Linked List, Examples: Image viewer, music player list etc. (to be used to explain concept of list). DLL node, List Operations – Create, appending nodes, delete, search.		1. Implement DLL. 2. Implement CDLL

			The Circular Linked List-Organization, List Operations – Appending nodes, delete, iterating circular list.		
8	01, 02, 03, 04	1, 2, 3, 4	Last In First Out (Stack) Data structures – Example: Reversing a word, evaluating an expression, message box etc. (to be used to explain concept of LIFO). The Stack implementation – push, pop, display. Stack Applications- Balanced Delimiters, Evaluating Postfix Expressions.		1. Implement Stack Data Structure. 2. Implement bracket matching using stack.
9	01, 02, 03, 04	1, 2, 3, 4,	Recursion. Properties of Recursion. Recursive functions: Factorials, RecursiveCall stack, The Fibonacci Sequence. How Recursion Works- The Run Time Stack. Recursive Applications- Recursive Binary Search, Towers of Hanoi.		1. Program to demonstrate recursive operations (factorial/ Fibonacci) 2. Implement solution for Towers of Hanoi.
10	01, 02, 03, 04	1, 2, 3, 4,	The First In First Out (Queue) Data structure – Example: Media player list, keyboard buffer queue, printer queue etc. (to be usedto explain concept of FIFO). Implementing the Queue and its operations using Python List. Priority Queues, Implementation.		1. Implement Queue. 2. Implement priority queue
11	01, 02, 03, 04	1, 2, 3, 4,	The Tree data structure – Example: File explorer/Folder structure, Domain name server. Tree Terminologies, Tree node representation. Binary trees, Binary search trees, Properties, Implementation of tree operations – insertion, deletion, search, Tree traversals (in order, pre order and post order).		1. Implement Binary search tree and its operations using list.
12	01, 02, 04	1, 2, 3, 4,	Depth-first traversal Breadth-first traversal Tree applications: Expression evaluation.		1. Implementations of BFS. 2. Implementation of DFS.
13	01, 03, 04	1, 2, 3, 4,	Introduction to Hashing. Hashing - Perfect hashing functions. Hash table Hash Functions, Operations, Hash collision, Application.		1. Implement Hash functions.
Total in hours			**39**	**13**	**52**

***PO = Program outcome as listed and defined in year 1 curriculum**

Table 1: Suggestive activities for tutorials (the list is only shared as an example and not inclusive of all possible activities for that course. Student and faculty are encouraged to choose activities that are relevant to the topic and the availability of such resources at their institution)

Sl. No	**Activity**
1	Design a Data structure for handling Student Records- Designing a Solution, Implementation (Using Basic DS).

2	Design a Data structure for handling Student Records- Designing a Solution, Implementation (Using ADT).
3	Optimize your solution (Bubble sort, selection sort and Insertion sort)
4	Implement Radix sort.
5	Prepare report on nonlinear data structures.
6	Design and implement sparse matrix representation using linked list.
7	Design and implement simple application that require DLL data structure.
8	Implement and demonstrate evaluating postfix expression.
9	Presentation on run time stack.
10	Design and implement priority queue data structure.
11	Prepare a Report on balanced trees.
12	Implement expression evaluation tree.
13	Prepare a report on hashing and analyze time complexity.

4. Reference:

Sl. No.	**Description**
1	Data Structures and Algorithms using Python by Rance D. Necaise
2	Python Data Structures and Algorithms by Benjamin Baka
3	www.geeksforgeeks.com

5. Equipment/software list

Sl. No.	**Particulars**	**Specification**	**Quantity**
1	Computers		20
2	Python 3.6		20
3	Editor such as iPython, Jupyter, spider, PyCharm		20

Operating System and Administration

Subject code – CSE402

1. Rationale

The Operating System knowledge and skill is an integral part in the study of computer science. It provides the platform for all other application to run on the machine, thus knowledge of operating system and administration becomes indispensable for understanding computing environment. It is essential to have knowledge of operating system's services and utilities to develop, deploy and maintain the software and hardware. The students will also be skilled in operating system virtualization, to create and manage virtual computing environment.

2. Course Outcomes: At the end of the course, the student will be able to:

CO-01	Explain functions and services of an operating system.
CO-02	Create a virtual environment and configure it to meet a specific application requirement.
CO-03	Identify and use Linux commands to create and manage simple file processing operations, organize directory structures, and develop shell script to automate given simple task.
CO-04	Demonstrate the role and responsibilities of a Linux system administrator and analyseproblems using suitable diagnostic tools and resolve issues.

3. Course Content

Week	CO	PO	Lecture (Knowledge Criteria)	Tutorial (Activity Criteria)	Practice (Performance Criteria)
1	1	1,7	Overview of Operating System, Need for OS, Structure, OS Types, Examplesof OS (desktop and mobile) Dual mode operation, Kernel and microkernel, Functions of OS User interfaces; Corporate Vs Personal needs; Types of OS installation	Refer Table 1	1. Types of OS installation 2. Boot methods 3 . File System and formatting 4. Post installation tasks
2	1, 2	1, 7	Virtualization technology, working, types		1. Install and configure

			Potentials and challenges of Virtualization, Virtual Machines, Containers. Linux Boot process. Linux command line - Interpreter, shell, CLI over GUI, Types of users- super and normal, Linux user manual.		virtual machine- Virtual box/VMware, VMware player station. 2. Download and install a terminal emulator and connect Linux VM via TE(optional). Significance of man command.
3	1,3	1, 7	File system - Pathnames, File system structure and its description, navigating the file system. File types, attributes, Access Control List (ACL), Adding text to file. Pipes, File Comparison, Filters / Text Processing Commands.	**Refer Table 1**	File and Directory commands: 1. Create and delete directories and files, File movement, copy commands, Pipes (named & unnamed) 2. Commands for viewing File, File comparison, File manipulation, Alteringfile permission, File compression and decompression. 3. Text processing commands.
4	1,	1,2,3,7	**Process Management** – Process, daemon, process states, PCB; Process scheduling Queue Operations on Processes - Process creation, Process termination, Interprocess communication. Scheduling - Long term, short term,and medium term; Context switch; Different types of CPU schedulers (Basic concept), Process priority; debugging (system hang)		1. Linux commands related to process creation and management- system calls fork() and exec(); bg, fg, nohup, pkill, nice, top, ps; 2. cron and at commands to schedule tasks.
5	1,3	1,2,3 4, 7	**Process synchronization-** critical section problem, Semaphores; Deadlock- System model, methods for handling deadlocks, deadlock prevention, avoidance, detection, recovery from deadlocks. Threads - Multithreading models,Threads, and processes. Types of threads - Kernel level and User level		1. Demonstration through videos. 2. Commands to exhibit thread concepts.
6	1	1,2,3 4, 7	**Memory management** - Process address space, static vs dynamic linking and loading.	**Refer Table 1**	1. Demonstration through videos. 2. Commands to view memory consumption

			Swapping, Memory allocation, Fragmentation, Paging, Segmentation; Virtual memory, Demand paging, Page replacement algorithm (concept only)		
7	1	2, 3 4, 7	**Shell Programming**: Basics of shell programming, types of shell in Linux, Basic Shell scripts- Shebang or Hashbang, Input & Output, decision making and iterative scripts.		1. Write shell scripts to illustrate decision making and different types of iterations; Ex- to perform string operations; to perform file operations;
8	1	2, 3 4, 7	**Automation of system tasks**: Writing scripts to automate commontasks.	**Refer Table 1**	1. Illustrate automation of basic tasks like monitoring memory consumption, check remote servers' connectivity, etc., at different frequencies.
9	1	2, 3,4	**Network Management** Network components- IP address, subnet mask, gateway. Network Interface management; Communication. Data transfer facilitation. Diagnosis and troubleshooting; Resource analysis.	**Refer Table 1**	1. Enable internet on Linux VM. 2. Test and manage network using following commands ifconfig, iwconfig, ethtool, arpwatch, bmon ,telnet, ssh, sendmail, mailstats, w cURL, wget, ftp, rcp, scp, rsync, sftp. netstat, ping, traceroute, iftop, nload, ss. tcpdump, dstat.
10	2,4	2,4,7	**User authentication** User and Group account management. Working on interface. Linux Directory Service - Account Authentication, what is LDAP and Active Directory? LDAP structure, working.	**Refer Table 1**	1. Work on user accounts useradd, passwd, userdel, usermod, groupadd, groupmod, gpasswd, groupdel; system-config. 2. OpenLDAP Installation 3. LDAP server and client configuration.
11	4	2, 3 4, 7	System monitoring, Log monitoring System maintenance, System information. System architecture, Linux Boot process and System run levels, System updates and repositories.		1. System monitoring commands top, df, dmesg, iostat 1, free, cat /proc/cpuinfo, cat/proc/meminfo; 2. Work on log directory - /var/log; 3. System maintenance commands- shutdown, reboot,

					halt, init. 4. System update & repositories- yum & rpm.
12	2,4	2, 3, 4, 7	Server setup: DNS- Introduction, Configuration, creating DNS zone, using DNS tools; FTP- Installation process, configuration and securing; setting up an Apache Web Server*(http)*	**Refer Table 1**	**Install and configure:** 1. DNS server with a domain name of your choice. 2. FTP server on LINUXand transfer files to demonstrate it's working. 3. Apache web server and create virtual hosts.
13	2,4	2, 3 4, 7	**Storage management:** Disk partition, formatting, mounting; Logical Volume Management (LVM)- Use of LVM, creating Volume groups, logical volume and disk mirroring, Extend Disk using LVM, Adding Swap Space Introduction to RAID – Hardware & Software, RAID levels.	**Refer Table 1**	1. Basic commands for storage partitions. 2. Install and configure LVM. 3. Add Disk and CreateStandard & LVM Partition. 4. Add virtual disk and create a new LVM partition(pvcreate, vgcreate, lvcreate) 5. Extend disk using LVM
Total in hours			**39**	**13**	**52**

***PO = Program outcome as listed and defined in year 1 curriculum**

Table 1: Suggestive activities for tutorials (the list is only shared as an example and not inclusive of all possible activities for that course. Student and faculty are encouraged to choose activities that are relevant to the topic and the availability of such resources at their institution)

1	1. Compare features of different OS(windows, Linux, RTOS- Vxworks/android) 2. Study the evolution of OS to recognize the importance of current OS trends. 3. Explain the different flavors of LINUX.
2	1. Explain OS level virtualization and state its benefits. 2. Compare VMs and Containers 3. Identify the difference between hypervisors and Linux containers. 4. Comprehend the benefits of virtualization.
3	1. Compare ex2/ex3 filesystem attributes. 2. Discuss the file- mount and unmount system calls.
4	**1.** Compare Linux fork () and Windows createprocess () functions.
5	1. Study probable conditions for deadlock occurrence and how to overcome it. 2. Identify relationship between threads and processes. 3. Comprehend the differences between types of threads
6	1. Compare the features of swapping and paging.
7	**1.** Compare different Linux shells.
8	1. Write a cron job that runs all essential apps. on an hourly/ daily/weekly/monthly basis. (for ex. Executing Antivirus)
9	1. Compare static and DHCP IP addresses and check whether these can be switched over. 2. Study different options offered by Linux for package management.
10	1. Identify few alternatives to openLDAP and make a comparison.
11	1. Explore other network commands required for a sysadmin and interpret their functions and usage.

12	1. Study the difference between application server and web server. 2. Identify the role of virtual host. 3. Explain different types of Apache virtual hosts and how they are set up.
13	1. Compare the features between RAID and SSD.

4. Reference:

Sl. No.	Description
1	Operating System internal and Design Principles, William Stallings
2	Operating System, Garry Nut
3	https://www.redhat.com/en/topics/virtualization
4	Virtual Machine - an overview | ScienceDirect Topics
5	DNS: https://www.youtube.com/watch?v=TiWs9n4fhys&list=RDCMUCQSpnDG3YsFNf5-qHocF-WQ&index
6	Linux system admin requirements: https://www.temok.com/blog/linux-system-administration/
7	Linux commands for modern sysadmins- N/W related - https://www.ubuntupit.com/useful-linux-network-commands-for-modern-sysadmins/
8	DNS Technology: https://www.digitalocean.com/community/tutorials/an-introduction-to-dns-terminology-components-and-concepts
9	Commands for Disk Management: https://www.programmersought.com/article/55913754022/

5. Equipment/software list

Sl. No.	Particulars	Specification	Quantity
1	Computers		
2	VirtualBox, Ubuntu or any other Linux OS image.		

Object Oriented Programming and Design with Java

Subject code – CSE403

1. Rationale

Object oriented programming paradigm with object-oriented design principles are vital in design and development of today's complex computing solutions. OOD principles provide valuable standards and guidelines to create clean and modular design and avoid code smells. Java being the popular object-oriented programming language that empowers the innovation in this digital world, students will have sound knowledge of object-oriented programming concepts and design principleswith java.

2. Course Outcomes: At the end of the course, the student will be able to:

CO-01	Design a solution for a given problem using object-oriented programming concepts and apply all appropriate object-oriented design principles
CO-02	Write and test the code for a designed solution using java OOP concepts.
CO-03	Identify exceptions in the designed or given solution and explain how to resolve them.
CO-04	Demonstrate with an example a java application's connection with a database.

3. Course Content

Week	**CO**	**PO***	**Lecture (Knowledge Criteria)**	**Tutorial (Activity Criteria)**	**Practice (Performance Criteria)**
1	1,2	1, 4	Introduction to Java Brief history; features; java architecture; components: JVM, JRE,JDK; Applications; Java environment setup; Structure of java program; Compilation and execution of javaprogram; Clean coding in java.	**Refer Table 1**	1. Install and Setup java environment 2. Install java editor (Eclipse for Enterprise Java) and configure workspace 3. Execution of first java program 4. Java code execution process
2	1,2	1, 2, 3, 4, 7	Introduction to OOP: Building blocks: class, object, attributes, methods; Class and objectsin java;		**1. Code, execute and debug programs** that uses different types of variables and datatypes;

			Variable: Types (local, instance, static); declaration, initialization; comments; 'Data types;		2. Identify and resolve issues in the given code snippet
3	1,2	1, 2, 3, 4	Constructors: rules for defining constructor; types; Destructor; Access modifiers; **this'** keyword; Autoboxing and unboxing; Operators;Expressions; Evaluation of expressions;		**1. Code, execute and debug programs** a. that uses different types of constructors b. for expression evaluation c. to perform autoboxing and unboxing 2. Identify and resolve issues in the given code snippet
4	1,2	1, 2, 3, 4, 7	**Memory allocation in java**; garbage collection: concept, working, types, advantages finalize () method;		1. Install memory monitoring tool and observe how JVM allocates memory 2. Memory allocation explanation through the programs
5	1,2	1, 2, 3, 4	**Conditional and Iterative statements** Decision making: if, if..else, switch Iterative: need of iterative statements; types of loops in java; how to use them; Break and continue statements;		1. Code, execute and debug programs that uses different control statements. 2. Identify and resolve issues in the given code snippet
6	1,2,3	1, 2, 3, 4, 7	**OOP concepts**: **Encapsulation** Concept; What is encapsulation? How to achieve encapsulation in java; Packages; **Single Responsibility Principle**: Intent; Rules; Benefits; example		1. Code, execute and debug programs 2. that uses encapsulation concept. 3. Define class & implement like simple calculator or text processing and check compliance with SRP.
7	1,2	1, 2, 3, 4	**Arrays**: Why arrays? Features, types, Declaration, array creation with new operator, working with arrays; Strings: creation, string methods;		1. Code, execute and debug programs that uses array concept 2. Code, execute and debug programs to perform string manipulation.
8	1,2	1, 2, 3, 4, 7	**OOP concepts**: **Inheritance** Inheritance concept; types; Inheritance in java; Examples; **Open Closed principle:** Intent; Rules; Benefits; example	**Refer Table 1**	1. Code, execute and debug programs that uses inheritance concept 2. Design a class & implement like file parser and check compliance with OCP.

9	1,2	1, 2, 3, 4,7	**OOP concepts**: **Polymorphism** Polymorphism concept; types: method overloading and overriding; application; polymorphism in java; sufficient examples;		1. Code, execute and debug programs that uses a. static binding b. dynamic binding
10	1,2	1, 2, 3, 4, 7	**OOP concepts: Abstraction** Overview; implementation of abstraction in java: abstract class and interface; Relationship between class and interface; inheritance in interface;Examples to substantiate the understanding of concepts; Eg. File parser; message logger		1. Code, execute and debug programs that uses 2. abstract class to achieve abstraction 3. interface to achieve abstraction 4. Verify whether the given code snippet is correct according to abstraction or not
11	1,2,3	1, 2, 3, 4, 7	**Files and Exception handling** Files and I/O streams: File reader and writer; Exception concept; exceptions in java; classification: checked and unchecked; exception handling in java;		**1. Code, execute and debug programs in java to** a. handles checked and unchecked exceptions b. read the content of the file and write the content to another file 2. Incorporate exception handling in programs/applications developed in previous sessions.
12	1,2,3	1, 2, 3, 4, 7	**Design principle:** **Interface Segregation principle:** Intent; Rules; Benefits; examples; Enums; Overview of java annotations;		1. Design an interface & implement it like one that builds different types of toys and check compliance with ISP.
13	1,2,3,4	1, 2, 3, 4, 7	**Database Connectivity** Introduction to JDBC; JDBC components; How JDBC works? JDBC connections; Connect java applicationto database using JDBC;	**Refer Table 1**	1. Code, execute and debug programs to connect to database through JDBC and perform basic DB operations
Total in hours			**39**	**13**	**52**

***PO = Program outcome as listed and defined in year 1 curriculum**

Table 1: Suggestive activities for tutorials (the list is only shared as an example and not inclusive of all possible activities for that course. Student and faculty are encouraged to choose activities that are relevant to the topic and the availability of such resources at their institution)

Sl. No	**Activity**

1	1. Identify various java IDEs and identify differences between them. 2. Compare and contrast Java with Python
2	1. Study and present a. type casting in java b. what are command line arguments in java? c. java keywords and their usage
3	1. Compare and contrast a. method and constructor; b. constructor and destructor
4	1. Study and present how does bytecode work in java.
5	1. Present nesting of conditional and iterative statements considering a use case.
6	Identify advantages and disadvantages of a. Encapsulation. b. Inheritance c. Abstraction d. Polymorphism
7	Study and report a. java Arrays class their methods b. java String class their methods
8	Identify and document how these principles help to avoid code smells. a. SRP b. OCP c. ISP
9	Compare and contrast a. static and dynamic binding and identify usage of each b. abstract class and interface, identify usage of each
10	1. Differentiate error and exception 2. Identify and document system exceptions
11	Study DRY principle, identify the benefits.
12	Identify how OOD principles violations impact the quality of code.
13	Identify java ORM frameworks and their features.
14	Study and find the inclusions in latest java versions.

4. Reference:

Sl. No.	Description
1	https://docs.oracle.com/javase/tutorial/java/concepts/
2	www.edureka.co
3	Clean Code by Robert C Martin
4	https://www.javabrahman.com/programming-principles/
5	https://medium.com/

5. Equipment/software list

Sl. No.	**Particulars**	**Specification**	**Quantity**
1	Computers		
2	Java 8.0 and above, eclipse		

Software Engineering Principles and Practices

Subject code – CSE404

1. Rationale

Digital reality has become an integral part of human life with software tools being used to deal with virtually every part of life. A process is key to develop a quality software successfully. Principles and practices of software engineering blends engineering, computing, project management and software development. It's essential to understand the life cycle of software development and the process followed to develop a quality software. Design thinking methodology encourages identifying alternative strategies and solutions to solve a problem in best possible way.

2. Course Outcomes: At the end of the Course, the student will be able to:

CO-01	Explain the typical software development life cycle (SDLC), list and differentiate the various SDLC models along with identifying where each model could be beneficial when applied.
CO-02	Demonstrate the application of design thinking as a process, explain how it helps in requirement engineering and mitigate risks.
CO-03	Study a given application requirement, create user stories, draw the appropriate UML diagram and validate to ensure user story/UML diagram meet with the given requirement.
CO-04	Document standard test procedures and test cases for a given requirement to ensure the software gives the desired results for which it is built.

3. Course Content

Week	CO	PO	Lecture (Knowledge Criteria)	Tutorial (Activity Criteria)	Practice (Performance Criteria)
1	1	1	**Overview** Software engineering; Need of software engineering; Software paradigms; Software product types: generic, customized; characteristics of good software; Challenges in software projects; Factors that influence software development; understanding success Software process; need of process, components of process, process	**Refer Table 1**	1. Discuss success and failure stories 2. Presentation of collected case studies 3. Enact the importance of ethical practices

			activities; Differentiate product, project and process; process assessment and improvement; Software engineering ethics.		
2	1	1, 5	**SDLC and Process Models** SDLC; Software process model; How to choose process model? Comparison between a defined process and an empirical process; Traditional process models: waterfall; Incremental; **Agile process-** manifesto; principles; practices; A paradigm shift from plan driven mentality.		1. Case study to understand the SDLC 2. Organize and play games to understand the agile process like, morning wake up game ▪ the marshmallow challenges ▪ White Elephant Sizing ▪ Easter Egg Challenge 3. Create JIRA (similar tool) account and learn interface
3	1	1, 5	Agile frameworks; Ceremonies; Roles; Overview of XP – XP practices Scrum: Overview; framework; ceremonies and artifacts		1. Play and act agile ceremonies 2. Play different agile roles Eg. Product owner, business analyst
4	1,2	1	**Risk** Risk, characteristics, categories; why risk management is critical; risk management framework; Activities; Principles of risk management, Risk identification, Risk assessment – risk analysis; risk prioritization; Risk Mitigation; need and importance of risk mitigation; Risk Control – planning, resolution, monitoring; How to use tool to manage and mitigate risks in an organization.		1. case study to understand the importance of risk management and mitigation of risk 2. How to use tools to manage and mitigate risks [eg. Logicgate, AuditBoard etc]
5	2	1, 2	**Design Thinking** Introduction, 5 stages of design thinking Understand the process of design thinking using an example Case Study		1. Conduct warmup activities to Ignite Design Thinking 2. Organize and conduct design thinking exercises and games

6	1,3	1, 2	**Requirement Engineering & Modeling** Overview; what is requirement? Importance; Requirement types; Sources ofrequirements; Requirement engineering Process; Feasibility study; Typical Requirements Engineering Problems; Requirement modeling strategies; Overview of UML; types of diagrams; **Note: Take a case study to understand requirement engineering and prepareuse cases or user stories**		1. Organize role play for requirement activities 2. Identify a problem and prepare requirement document or Epics and user stories 3. Configure JIRA for the managing the project to solve the identified problem 4. Draw UML diagram for given use case
7	1,3	1, 2, 4	**User stories** What are user stories? Why user story? Basic concepts; Characteristics; How to write/create user stories? Steps; 3C's in user stories; Life cycle of user story. User story map. **Estimation:** User story point: basics; components of story point estimation; Steps involved in estimation;		1. Create detailed user stories for theabove identified problem 1.Organize and play planning poker to decide on user points.
8	1,2	1, 2, 3	**Design** Objectives; design Concepts; Levels of design; Architectural styles; Monolithic and Microservices; UI and UX: Overview of UI and UX, UI types,essential properties, elements of UI design;relationship between UI and UX; Importance of good UI/UX. Wireframes: overview, purpose, benefits;		1.Create sitemap and wireframe for above created user stories. (Tools such as sketch, Adobe XD, Figma, etc can be used)
9	1	1, 2, 3, 4	**Development** Overview of DevOps; working principle; Benefits; DevOps culture; DevOps practices: continuous integration, continuous delivery, version control, configuration management, Build process;		1. Create Git (similar tool) account and configure repository 2. Upload the artifacts created toGit Learn version control and configuration management with Git
10	1	1, 2, 3, 4	**Code quality and code security**: overview; importance; issues caused by poor code; tools to check code quality **Containerization**: Container, why container, containerization; working principle; benefits; Hello world example **Note: Docker or similar tool can be used to explain the containers.**		1. Install and configure Jenkins 2. Create a container image for Hello world project 3.Setup build for container image using Jenkins (Hello world application)

11	1,4	1, 4	**Testing** Principles of testing; Need of testing; stages; Testing process and activities; classification; Testing strategies; Levels of software testing; Software testing types; (Integration testing, functional testing, end-to-end testing need to be explained in detail)		1. Prepare Test plan for the user stories using JIRA 2. Prepare RTM for the user stories created using JIRA. Create test cases for the user stories created.
12	1,4	1	**Software Measurement and Metrics** Measurement; need of Measurement; types; Metrics: characteristics; classification; Agile metrics; Application monitoring.		1. Use JIRA or similar tool to capture agile metrics 2. Use SonarQube to capture code quality metrics
13	1,4	1	**Quality Control and Assurance** Concept of software quality, Compliance, Quality Standards, quality control, quality assurance; Difference between QC & QA. Need for auditing. Auditing fundamentals: auditing, elements of auditing; audit types; auditing methods, benefits of auditing. Quality and Process improvement tools and techniques– pareto chart, PDCA cycle, Six sigma and Lean process		1. Organize Roleplay to understand the roles and responsibilities of QA and QC team. 2. Audit the artifacts produced in previous sessions
Total in hours			39	13	52

***PO = Program outcome as listed and defined in year 1 curriculum**

Table 1: Suggestive activities for tutorials (the list is only shared as an example and not inclusive of all possible activities for that course. Student and faculty are encouraged to choose activities that are relevant to the topic and the availability of such resources at their institution)

Sl No	**Activity**
1	Study the traffic signal and the importance of rules and process.
2	Visit various consulting company web portals and collect case studies.
3	Document the roles and responsibilities of different agile ceremonies
4	Identify cost of risk; Identify commonly used risk management tools.
5	Identify a problem and explain how design thinking can be applied to solve it. Design a shopping cart to achieve ease of use, applying design thinking.
6	Prepare RPM requirement traceability matrix for shopping cartList the criteria to select the requirement management tools. Identify different requirement management tools and list their features. Identify frequently used UML diagrams and also identify tools used to draw them.
7	Explore agile estimation techniques and prepare a report.
8	Study boiler plate and present necessary characteristics of boiler plate for a large and small project
9	Identify different DevOps Tools and list their features Study and report OWASP coding guidelines Learn and report Twelve Factor App methodology Identify different version control and configuration management tools and report their marketshare

10	Compare and contrast containerization and virtualization and identify importance of these in software development Identify container providers
11	Study and prepare report on testing tools. Compare manual and automation testing
12	Study and prepare report on widely used software metrics.
13	Identify different quality tools and report their features and usage

4. Reference:

Sl. No.	Description
1	Agile Software Development, principles, patterns and practices by Robert Martin
2	Art of agile development by James Shore & Shane Warden
3	Extreme programming explained: embrace change
4	Software-Engineering-9th-Edition-by-Ian-Sommerville
5	RPL-7th_ed_software_engineering_a_practitioners_approach_by_roger_s._pressman_
6	Becoming Agile..in an imperfect world by Greg Smith, Ahmed Sidky
7	scaledagileframework.com
8	Continuous Delivery Principles | Atlassian
9	www.agilealliance.org/
10	www.udemy.com
11	www. tutorialride.com
12	www.interaction-design.org/
13	www.digite.com

5. Equipment/software list

Sl. No.	Particulars	Specification	Quantity
1.	Computers		
2.	Git, Jira, SonarCube, Lucidchart or any other UML design tool		