AC – 27/12/2023
Item No. – 6.8 (N)

As Per NEP 2020

University of Mumbai

Title of the program

A- U.G. Certificate in **Computer Science**
B- U.G. Diploma in **Computer Science**
C- B.Sc. (**Computer Science**)
D- B.Sc. (Hons.) in **Computer Science**
E- B.Sc. (Hons. with Research) in **Computer Science**

Syllabus for

Semester – I & II

Ref: GR dated 20th April, 2023 for Credit Structure of UG

(With effect from the academic year 2024-25 progressively)

University of Mumbai

(As per NEP 2020)

Sr. No.	Heading	Particulars	
1	**Title of program** **O: _____A**	**A**	**U.G. Certificate in Computer Science**
	O: _____B	**B**	**U.G. Diploma in Computer Science**
	O: _____C	**C**	**B.Sc. (Computer Science)**
	O: _____D	**D**	**B.Sc. (Hons.) in Computer Science**
	O: _____E	**E**	**B.Sc. (Hons. with Research) in Computer Science**
2	**Eligibility** **O: _____A**	**A**	A candidate for being eligible for admission must have passed **Higher Secondary School Certificate Examination** (Std. XII) in **Science stream** conducted by the Maharashtra State Board of Secondary and Higher Secondary Education with **Mathematics and Statistics as one of the subject** or its equivalent. Admission will be on merit, based on order of preference as follows: 1. Aggregate Marks at H.S.C. or equivalent. 2. Aggregate Marks in Science Group (Physics, Chemistry and Mathematics) 3. Marks in Mathematics and Statistics and Physics. Marks in Mathematics and Statistics. **OR** Passed Equivalent **Academic Level 4.0** with **Mathematics and Statistics as one of the subject**
	O: _____B	**B**	Under Graduate Certificate in Computer Science **OR** Passed Equivalent **Academic Level 4.5**
	O: _____C	**C**	Under Graduate Diploma in Computer Science **OR** Passed Equivalent **Academic Level 5.0**
	O: _____D	**D**	Bachelors of Science in Computer Science with minimum CGPA of 7.5 **OR** Passed Equivalent **Academic Level 5.5**

	O: _____E	**E**	Bachelors of Science in Computer Science with minimum CGPA of 7.5 **OR** Passed Equivalent **Academic Level 5.5**
3	**Duration of program** **R: _____**	**A**	One Year
		B	Two Years
		C	Three Years
		D	Four Years
		E	Four Years
4	**Intake Capacity** **R: _____**	**60 students per division**	
5	**Scheme of Examination** **R: _____**	NEP **40% Internal** **60% External**, Semester End Examination **Individual Passing** in Internal and External Examination	
6	**Standards of Passing** **R: _____**	**40%** in **each component**	
7	**Credit Structure** **Sem. I - R: _____A** **Sem. II - R: _____ B**	Attached herewith	
	Credit Structure **Sem. III - R: _____C** **Sem. IV - R: _____D**		
	Credit Structure **Sem. V - R: _____E** **Sem. VI - R: _____F**		
8	**Semesters**	**A**	Sem I & II
		B	Sem III & IV
		C	Sem V & VI
		D	Sem VII & VIII
		E	Sem VII & VIII
9	**Program Academic Level**	**A**	4.5
		B	5.0
		C	5.5
		D	6.0
		E	6.0
10	**Pattern**	Semester	
11	**Status**	New	
12	**To be implemented from Academic Year Progressively**	From Academic Year: 2024-25	

This syllabus is applicable to IDOL students as well, w. e. f. 2025-26.

Sign of the BOS Chairman	**Sign of the Offg. Associate Dean**	**Sign of Offg. Dean**
Dr. Jyotshna Dongardive	**Dr. Madhav R. Rajwade**	**Prof. Shivram S. Garje**
Ad-hoc BOS (Computer Science)	Faculty of Science & Technology	Faculty of Science & Technology

Preamble

1) Introduction

In the era of Information and Communication Technology (ICT), the transformative impact of computers on society is undeniable. The pervasive applications of computing across diverse fields have given rise to dynamic industries, evolving in tandem with the swift pace of technological change. As the landscape of the computing field continues to advance, it becomes imperative for students to cultivate a robust foundation that not only facilitates their current skills but also empowers them to adapt to the evolving nature of the field.

In line with the National Education Policy (NEP) 2020, our revised Computer Science program is designed to instill in students the ability to navigate the ever-changing technological terrain. Recognizing that specific languages and platforms may undergo transformations, the curriculum places a strong emphasis on fostering adaptability. Students will not only be exposed to a diverse array of programming languages, tools, paradigms, and technologies but will also delve into the fundamental principles that underpin the realm of computer science.

The core of our program encompasses essential courses such as programming languages, data structures, computer architecture and organization, algorithms, database systems, operating systems, and software engineering. Complementing these foundational elements are specialized courses in areas such as artificial intelligence, computer-based communication networks, distributed computing, information security, graphics, human-computer interaction, multimedia, scientific computing, web technology, and other cutting-edge topics in computer science.

Key Philosophy of the Program:

- **Form Strong Foundations:** Lay the groundwork for a comprehensive understanding of Computer Science.
- **Nurture Skills:** Develop programming, analytical, and design skills to tackle real-world problems effectively.
- **Introduce Gradually:** Familiarize students with emerging trends in a gradual and coherent manner.
- **Prepare for Industry Challenges:** Groom students to meet the challenges of the ICT industry with confidence and competence.

In acknowledgement of the evolving aspirations of students, our program not only prepares them for careers in the industry but also opens doors to research opportunities. The primary goal is to deliver a modern curriculum that equips graduates with both theoretical depth and practical acumen, empowering them to excel in the workplace while fostering a mindset of lifelong learning.

This program not only paves the way for a successful career in the software industry but also inspires students to pursue further studies and research opportunities. Graduates can seamlessly transition into postgraduate programs in Computer Science, leading to research and development roles, employment in IT industries, or even a career in business management.

As we unveil this syllabus, we invite students on a journey of exploration, learning, and innovation, ensuring they are not only prepared for the present but also poised to shape the future of Computer Science.

2) Aims and Objectives

Understanding and Knowledge Base: Develop a profound understanding and knowledge of the fundamental theories, systems, and applications that form the bedrock of Computer Science. This includes establishing a strong foundation in theoretical concepts and cultivating expertise in the practical application of Computer Science theories.

Analytical Abilities and Problem Solving: Foster essential skills and analytical abilities required for devising computer-based solutions to real-life problems. This involves developing critical thinking skills for problem identification and analysis, as well as cultivating the ability to design and implement effective solutions using computational tools.

Training in Emerging Technologies: Provide training in emergent computing technologies, facilitating the development of innovative solutions for both industry and academia. This includes exposing students to cutting-edge technologies and their applications, as well as encouraging exploration and experimentation with emerging tools and platforms.

Preparation for Post-Graduate Studies: Develop the necessary study skills and knowledge for students to pursue further post-graduate study in Computer Science or related fields. This involves equipping students with the academic rigor required for advanced studies and fostering a passion for continuous learning and research in the field.

Professional Skillset Development: Develop the professional skillset required for a successful career in an information technology-oriented business or industry. This includes providing practical exposure to industry-relevant tools and practices, as well as instilling a sense of professional ethics and responsibility.

Independent and Collaborative Work: Enable students to work independently and collaboratively, communicate effectively, and become responsible, competent, confident, insightful, and creative users of computing technology. This involves cultivating independence in problem-solving and project execution, as well as enhancing communication and collaboration skills for effective teamwork.

3) Learning Outcomes

At the end of three year Bachelor of Computer Science the students will be able:

- Formulate, model, and design solutions and procedures, utilizing software tools to address real-world problems effectively.
- Design and develop computer programs and computer-based systems in diverse areas such as networking, web design, security, cloud computing, IoT, data science, and other emerging technologies.
- Familiarize themselves with modern-day trends in industry and research-based settings, fostering the ability to innovate novel solutions to existing problems.
- Apply concepts, principles, and theories related to computer science to new and challenging situations.
- Demonstrate proficiency in using current techniques, skills, and tools essential for computing practice.
- Apply standard Software Engineering practices and strategies in real-time software project development.
- Pursue higher studies of specialization and confidently enter technical employment.
- Work independently or collaboratively as effective team members on substantial software projects, showcasing project management and teamwork skills.
- Communicate and present their work effectively and coherently, both in oral and written formats.
- Display ethical conduct in the usage of the Internet and Cyber systems, understanding and adhering to ethical standards in computing practices.
- Engage in independent and life-long learning, adapting to the rapidly changing IT industry and staying abreast of evolving technologies.

4) Credit Structure of the Program (Sem I, II, III & IV) (Table as per Parishisht 2 with sign of HOD and Dean)

Under Graduate Certificate in Computer Science

	R:_____A									
Level	**Semester**	**Major**		**Minor**	**OE**	**VSC, SEC (VSEC)**	**AEC, VEC, IKS**	**OJT, FP, CEP, CC, RP**	**Cum. Cr. / Sem.**	**Degree/ Cum. Cr.**
		Mandatory	**Electives**							
4.5	**I**	**MJ1:** Digital Systems & Architecture (TH) – 2 **MJ2:** Fundamentals of Database Systems (TH) – 2 **MJP1:** Computer Science Practical 1 (PR) – 2 **6**	**-**	**-**	2+2	**VSC:2** Introduction to Programming with Python – 2 **SEC:2** Statistics with R Programming – 2 **OR** Linux Operating System – 2	AEC:2, VEC:2, IKS:2	CC:2	22	**UG Certificate 44**
	R:_____B									
	II	**MJ3:** Design & Analysis of Algorithms (TH) – 2 **MJ4:** Object Oriented Programming (TH) – 2 **MJP2:** Computer Science Practical 2 (PR) – 2 **6**	**-**	2	2+2	**VSC:2** Web Technologies – 2 **SEC:2** Database Management Systems using PL/SQL – 2 **OR** Advanced Python Programming – 2	AEC:2, VEC:2	CC:2	22	
	Cum Cr.	**12**	**-**	**2**	**8**	**4+4**	**4+4+2**	**4**	**44**	
Exit option: Award of UG Certificate in Major with 40-44 credits and an additional 4 credits core NSQF course/ Internship OR Continue with Major and Minor										

Under Graduate Diploma in Computer Science

	R:_____C									
Level	**Semester**	**Major**		**Minor**	**OE**	**VSC, SEC (VSEC)**	**AEC, VEC, IKS**	**OJT, FP, CEP, CC, RP**	**Cum. Cr. / Sem.**	**Degree/ Cum. Cr.**
		Mandatory	**Electives**							
5.0	**III**	**MJ5:** Principles of Operating Systems (TH) – 2 **MJ6:** Theory of Computation (TH) – 2 **MJ7:** Data Structures (TH) – 2 **MJP3:** Computer Science Practical 3 (PR) – 2 **8**	-	4	2	**VSC:2** Java Programming – 2	AEC:2	FP: 2CC:2	22	**UG Diploma 88**
	R:_____D									
	IV	**MJ8:** Computer Networks (TH) – 2 **MJ9:** Software Engineering (TH) – 2 **MJ10:** IoT Technologies (TH) – 2 **MJP4:** Computer Science Practical 4 (PR) – 2 **8**	-	4	2	**SEC:2** Mobile Application Development – 2 **OR** MEAN Stack Development – 2	AEC:2	CEP: 2 CC:2	22	
	Cum Cr.	**28**	**-**	**10**	**12**	**6+6**	**8+4+2**	**8+4**	**88**	
Exit option; Award of UG Diploma in Major and Minor with 80-88 credits and an additional 4 credits core NSQF course/ Internship OR Continue with Major and Minor										

B.Sc. (Computer Science)

Level	Semester	Major		Minor	OE	VSC, SEC (VSEC)	AEC, VEC, IKS	OJT, FP, CEP, CC, RP	Cum. Cr. / Sem.	Degree/ Cum. Cr.
		Mandatory	Electives							
	R:_____E									
5.5	**V**	**MJ11:** Artificial Intelligence (TH) – 2 **MJ12:** Cyber & Information Security (TH) – 2 **MJ13:** Moral & Ehtical AI (TH) – 2 **MJP5:** Computer Science Practical 5 (PR) – 2 **MJP6:** Mini Project – I (PR) – 2 **10**	**MJEL1:** Software Testing & Quality Assurance (TH) – 2 **OR** **MJEL2:** Wireless & Sensor Networks (TH) – 2 **MJELP1:** Software Testing & Quality Assurance Practical (PR) – 2 **OR** **MJELP2:**Wireless & Sensor Networks Practical (PR) – 2 **4**	4	-	**VSC: 2** Ethical Hacking – 2	-	FP/CEP:2	22	**UG Degree 132**
	R:_____F									
	VI	**MJ14:** Data Science (TH) – 2 **MJ15:** Cloud Computing (TH) – 2 **MJ16:** Software Project Management (TH) – 2 **MJP7:** Computer Science Practical 6 (PR) – 2 **MJP8:** Mini Project – II (PR) – 2 **10**	**MJEL3:** Information Retrieval (TH) – 2 **OR** **MJEL4:** Linux Server Administration (TH) – 2 **MJELP3:** Information Retrieval Practical (PR) – 2 **OR** **MJELP4:** Linux Server Administration Practical (PR) – 2 **4**	4	-	-	-	OJT:4	22	
	Cum Cr.	**48**	**8**	**18**	**12**	**8+6**	**8+4+2**	**8+6+4**	**132**	
Exit option: Award of UG Degree in Major with 132 credits OR Continue with Major and Minor										

[Abbreviation - OE – Open Electives, VSC – Vocation Skill Course, SEC – Skill Enhancement Course, (VSEC), AEC – Ability Enhancement Course, VEC – Value Education Course, IKS – Indian Knowledge System, OJT – on Job Training, FP – Field Project, CEP – Continuing Education Program, CC – Co-Curricular, RP – Research Project]

Semester I

Component	Major		Minor	OE	VSC	SEC	AEC	VEC	IKS	CC	Total
	Mandatory	Electives									
Credits	2+2+2	---	---	2+2	2	2	2	2	2	2	22

Component	Subject	Total Credits
Major	Digital Systems & Architecture	**2**
Major	Fundamentals of Database Systems	**2**
Major	Computer Science Practical 1	**2**
VSC	Introduction to Programming with Python	**2**
SEC (any one)	Statistics with R Programming	**2**
	Linux Operating System	

Semester II

Component	Major		Minor	OE	VSC	SEC	AEC	VEC	IKS	CC2	Total
	Mandatory	Electives									
Credits	2+2+2	---	2	2+2	2	2	2	2	---	2	22

Component	Subject	Total Credits
Major	Design & Analysis of Algorithms	**2**
Major	Object Oriented Programming using C++	**2**
Major	Computer Science Practical 2	**2**
VSC	Web Designing	**2**
SEC (any one)	Database Management Systems using PL/SQL	**2**
	Advanced Python Programming	

Sem – I

Mandatory Courses

Name of the Course: Digital System and Architecture

Sr. No.	Heading	Particulars
1	**Description the course:**	**Introduction:** The Digital Systems and Architecture course serves as a foundational exploration into the fundamental principles governing digital systems and computer architecture. This course delves into the design and organization of digital circuits and systems that form the backbone of modern computing devices. **Relevance:** In the era of rapid technological advancement, understanding digital systems and architecture is paramount. From smartphones to supercomputers, digital systems are pervasive. This course is essential for anyone aspiring to comprehend the inner workings of these systems and contribute to their development. **Usefulness:** The course equips students with the knowledge and skills to design, analyze, and optimize digital systems. It serves as a gateway for students to explore various aspects of computer architecture, laying the groundwork for more advanced studies and applications in the field. **Application:** Knowledge gained in this course finds practical applications in diverse domains, including embedded systems, computer networks, signal processing, and beyond. Students will learn how to translate theoretical concepts into tangible solutions, bridging the gap between abstraction and real-world implementation. **Interest:** Digital System and Architecture is an intellectually stimulating course that captivates students with its blend of theoretical concepts and hands-on application. The allure of creating efficient and high-performing digital systems often sparks curiosity and enthusiasm among students. **Connection with Other Courses:** This course establishes crucial linkages with other courses in computer science. It provides a solid

		foundation for more advanced courses such as computer organization, microprocessor systems, and hardware description languages. The knowledge gained here forms a seamless continuum in the study of computer systems. **Demand in the Industry:** As the demand for faster, more efficient computing systems continues to rise, professionals well-versed in digital systems and architecture are highly sought after. Industries ranging from electronics and telecommunications to automotive and healthcare actively seek individuals with expertise in designing and optimizing digital systems. **Job Prospects:** Graduates with proficiency in digital systems and architecture find themselves well-positioned for a myriad of career opportunities. Roles may include digital design engineer, embedded systems developer, hardware architect, and systems analyst. The skills acquired in this course open doors to a wide array of industries where digital technology plays a pivotal role.
2	**Vertical:**	Major
3	**Type:**	Theory
4	**Credits:**	2 credits
5	**Hours Allotted:**	30 Hours
6	**Marks Allotted:**	50 Marks
7	**Course Objectives(CO):** **CO 1.** To understand fundamentals of Logic gates, Number system and Flip Flops. **CO 2.** To have an understanding of Digital System and Operation of a Digital Computer. **CO 3.** To Learn Different Architecture & Organization of memory system, processor organization and control unit. **CO 4.** Basic understanding of 8085 microprocessor and its applications.	
8	**Course Outcomes (OC):** After successful completion of this course, students would be able to - **OC 1.** Learn how number system and codes are useful in computer system design. **OC 2.** Learn how Flip Flops are useful in memory design and data communication through CPU and Memory and I/O devices. **OC 3.** Learn about basics of instruction sets and its types. **OC 4.** Learn about Processor Internal Architecture and Design.	

9	**Modules:-** **Module 1 (15 hours):**
	Fundamentals of Digital Logic: Boolean algebra, Logic Gates, Simplification of Logic Circuits: Algebraic Simplification, Karnaugh Maps. **Combinational Circuits:** Adders, Subtractors, Multiplexer, De-Multiplexer. **Sequential Circuits:** Flip- Flops (SR, JK & D), Counters: synchronous and asynchronous Counter. **Computer System:** Comparison of Computer Organization & Architecture, Computer Components and Functions, Interconnection Structures. Bus Interconnections, Input / Output: I/O Module Programmed I/O, Interrupt Driven I/O, Direct Memory Access.
	Module 2 (15 hours):
	Memory System Organization: Classification and design parameters, Memory Hierarchy, Internal Memory: RAM, SRAM and DRAM, Interleaved and Associative Memory. Cache Memory: Design Principles, Memory mappings, Replacement Algorithms, Cache performance, Cache Coherence. Virtual Memory, External Memory: Magnetic Discs, Optical Memory, Flash Memories, RAID Levels **Instructions:** Instruction Formats, Instruction Sets, Addressing Modes, Addressing Modes Examples with Assembly Language [8085/8086 CPU]. **Processor Organization:** Structure and Function. Register Organization [8085/8086 CPU]. Basic Microprocessor operations: Data Transfer (Register / Memory) Operations, Arithmetic & Logical Operations. Instruction Cycle, Instruction Pipelining. Introduction to RISC and CISC Architecture, Instruction Level Parallelism and Superscalar Processors, Design Issues.
10	**Text Books** 1. M. Mano, Computer System Architecture 3rd edition, Pearson 2. Carl Hamacher et al., Computer Organization and Embedded Systems, 6 ed., McGraw-Hill 2012 3. R P Jain, Modern Digital Electronics, Tata McGraw Hill Education Pvt. Ltd. , 4th Edition, 2010
11	**Reference Books** 1. William Stallings (2010), Computer Organization and Architecture-designing for performance, 8th edition, Prentice Hall, New Jersy. 2. Anrew S. Tanenbaum (2006), Structured Computer Organization, 5th edition, PearsonEducation Inc, 3. John P. Hayes (1998), Computer Architecture and Organization, 3rd edition, Tata McGrawHill 4. Ramesh Gaonkar (2013), Microprocessor Architecture, Programming and Application with 8085, 6^{th} edition, Penram.

<table>
<tr><td>12</td><td>Internal Continuous Assessment: 40%</td><td>Semester End Examination: 60%</td></tr>
<tr><td>13</td><td>Continuous Evaluation through:
Class Test on Module 1: 10 marks
Class Test on Module 2: 10 marks
Average of 2 Class Tests: 10 marks
Assignment on Module 1: 5 marks
Assignment on Module 2: 5 marks
Total of 2 Assignments: 10 marks
Total: 20 marks</td><td>Evaluation through:
A Semester End Theory Examination of 1 hour duration for 30 marks as per the paper pattern given below.
Total: 30 marks</td></tr>
<tr><td>14</td><td colspan="2">Format of Question Paper:

Total Marks: 30 Duration: 1 Hour
<table>
<tr><th>Question</th><th>Based On</th><th>Options</th><th>Marks</th></tr>
<tr><td>Q. 1</td><td>Module 1</td><td>Any 2 out of 4</td><td>10</td></tr>
<tr><td>Q. 2</td><td>Module 2</td><td>Any 2 out of 4</td><td>10</td></tr>
<tr><td>Q. 3</td><td>Module 1 & 2</td><td>Any 2 out of 4</td><td>10</td></tr>
</table></td></tr>
</table>

Name of the Course: Fundamentals of Database Systems

Sr. No.	Heading	Particulars
1	**Description the course:**	**Introduction:** The Fundamentals of Database Systems course is a foundation in the study of information management and technology. It provides students with a comprehensive understanding of the principles, design, and implementation of databases, which are critical components in virtually every domain where data is utilized. **Relevance:** In today's data-driven world, the management and retrieval of information are paramount. This course is highly relevant as it addresses the core concepts essential for organizing, storing, and manipulating data efficiently. **Usefulness:** This course is immensely useful for individuals aspiring to work with data in various capacities. Whether designing databases, developing applications that interact with databases, or analyzing data trends, a solid understanding of database fundamentals is crucial. **Application:** The principles learned in this course find application across diverse sectors, including business, healthcare, finance, and technology. Students will gain the skills to model real-world scenarios, design efficient databases, and implement systems that store and retrieve information seamlessly. **Interest:** This course often attracts students due to its practical and tangible applications. The ability to structure and manage data effectively, ensuring its integrity and accessibility, can be intellectually stimulating and applicable to numerous real-world scenarios. **Connection with Other Courses:** This course forms a vital connection with various other courses in computer science and information technology. It is foundational to courses like database management, data warehousing, and data mining. Additionally, it complements courses related to software development, ensuring a holistic understanding of system architecture.

		Demand in the Industry: As businesses and organizations amass ever-growing volumes of data, there is an increasing demand for professionals versed in database systems. Industries such as finance, healthcare, e-commerce, and technology actively seek individuals who can design, implement, and manage robust databases. **Job Prospects:** Graduates proficient in the fundamentals of database systems enjoy promising job prospects. Potential roles include database administrator, data analyst, database developer, and business intelligence analyst. These professionals play a pivotal role in ensuring the efficient and secure management of an organization's data assets.
2	**Vertical:**	Major
3	**Type:**	Theory
4	**Credits:**	2 credits (1 credit = 15 Hours for Theory)
5	**Hours Allotted:**	30 Hours
6	**Marks Allotted:**	50 Marks
7	**Course Objectives(CO):** **CO 1.** To make students aware fundamentals of database system. **CO 2.** To give idea how ERD components helpful in database design and implementation. **CO 3.** To experience the students working with database using MySQL. **CO 4.** To familiarize the student with normalization, database protection and different DCL Statements. **CO 5.** To make students aware about importance of protecting data from unauthorized users. **CO 6.** To make students aware of granting and revoking rights of data manipulation.	
8	**Course Outcomes (OC):** After successful completion of this course, students would be able to - **OC 1.** To appreciate the importance of database design. **OC 2.** Analyze database requirements and determine the entities involved in the system and their relationship to one another. **OC 3.** Write simple queries to MySQL related to String, Maths and Date Functions. **OC 4.** Create tables and insert/update/delete data, and query data in a relational DBMS using MySQL commands. **OC 5.** Understand the normalization and its role in the database design process. **OC 6.** Handle data permissions. **OC 7.** Create indexes and understands the role of Indexes in optimization search.	

9	**Modules** **Module 1 (15 hours):**
	Introduction to DBMS: Database, DBMS – Definition, Overview of DBMS, Advantages of DBMS, Levels of abstraction, Data independence, DBMS Architecture **Data models**: Client/Server Architecture, Object Based Logical Model, Record Based Logical Model (relational, hierarchical, network **Entity Relationship Model and ER to Table:** Entities, attributes, entity sets, relations, relationship sets, Additional constraints (key constraints, participation constraints, weak entities, aggregation / generalization, Conceptual Design using ER (entities VS attributes, Entity Vs relationship, binary Vs ternary, constraints beyond ER) Entity to Table, Relationship to tables with and without key constraints. **DDL Statements:** Creating Databases, Using Databases, datatypes, Creating Tables (with integrity constraints – primary key, default, check, not null), Altering Tables, Renaming Tables, Dropping Tables, Truncating Tables **DML statements:** Viewing the structure of a table insert, update, delete, Select all columns, specific columns, unique records, conditional select, in clause, between clause, limit, aggregate functions (count, min, max, avg, sum), group by clause, having clause
	Module 2 (15 hours):
	Relational data model: Domains, attributes, Tuples and Relations, Relational Model Notation, Characteristics of Relations, Relational Constraints - primary key, referential integrity, unique constraint, Null constraint, Check constraint **Functions:** String Functions (concat, instr, left, right, mid, length, lcase/lower, ucase/upper, replace, strcmp, trim, ltrim, rtrim), Math Functions (abs, ceil, floor, mod, pow, sqrt, round, truncate) Date Functions(adddate, datediff, day, month, year, hour, min, sec, now, reverse) **Joining Tables and Subqueries**: inner join, outer join (left outer, right outer, full outer) subqueries with IN, EXISTS, subqueries restrictions, Nested subqueries, ANY/ALL clause, correlated subqueries **Normal forms:** Functional dependencies, first, second, third, and BCNF normal forms based on primary keys, lossless join decomposition. **Database Protection:** Security Issues, Threats to Databases, Security Mechanisms, Role of DBA, Discretionary Access Control, Backing Up and Restoring databases **Views**: Creating, altering dropping, renaming and manipulating views **DCL Statements:** Creating/dropping users, privileges introduction, granting/revoking privileges, viewing privileges), Transaction control commands – Commit, Rollback

<table>
<tr><td>10</td><td colspan="2">Text Books
1. Fundamentals of Database System, ElmasriRamez, NavatheShamkant, Pearson Education, Seventh edition, 2017
2. Database Management Systems, Raghu Ramakrishnan and Johannes Gehrke, 3rd Edition,2014
3. Murach's MySQL, Joel Murach, 3rd Edition, 3rd Edition, 2019</td></tr>
<tr><td>11</td><td colspan="2">Reference Books
1. Database System Concepts, Abraham Silberschatz, HenryF.Korth, S.Sudarshan, McGraw Hill,2017
2. MySQL: The Complete Reference, VikramVaswani , McGraw Hill, 2017
3. Learn SQL with MySQL: Retrieve and Manipulate Data Using SQL Commands with Ease, Ashwin Pajankar, BPB Publications, 2020</td></tr>
<tr><td>12</td><td>Internal Continuous Assessment: 40%</td><td>Semester End Examination: 60%</td></tr>
<tr><td>13</td><td>Continuous Evaluation through:
Class Test on Module 1: 10 marks
Class Test on Module 2: 10 marks
Average of 2 Class Tests: 10 marks
Assignment on Module 1: 5 marks
Assignment on Module 2: 5 marks
Total of 2 Assignments: 10 marks
Total: 20 marks</td><td>Evaluation through:
A Semester End Theory Examination of 1 hour duration for 30 marks as per the paper pattern given below.
Total: 30 marks</td></tr>
<tr><td>14</td><td colspan="2">Format of Question Paper:

Total Marks: 30 Duration: 1 Hour
<table>
<tr><th>Question</th><th>Based On</th><th>Options</th><th>Marks</th></tr>
<tr><td>Q. 1</td><td>Module 1</td><td>Any 2 out of 4</td><td>10</td></tr>
<tr><td>Q. 2</td><td>Module 2</td><td>Any 2 out of 4</td><td>10</td></tr>
<tr><td>Q. 3</td><td>Module 1 & 2</td><td>Any 2 out of 4</td><td>10</td></tr>
</table></td></tr>
</table>

Name of the Course: Computer Science Practical 1

Sr. No.	Heading	Particulars
1	**Description the course:**	**Introduction:** The Major Computer Science Practical Course, encompassing Digital Systems and Architecture as well as Database Systems, is a comprehensive and hands-on exploration into the foundational aspects of both hardware and software that underpin modern computing. This practical course is designed to provide students with a holistic understanding of digital systems, computer architecture, and the effective management of data within databases. **Relevance:** In an era where seamless integration of hardware and software is pivotal, the combination of Digital Systems and Architecture with Database Systems is highly relevant. This practical course addresses the symbiotic relationship between the two, offering students a holistic perspective on building robust computing solutions. **Usefulness:** This course is immensely useful for students aiming to bridge the gap between hardware and software. By integrating digital systems with database concepts, students gain a unique skill set that enables them to design, implement, and optimize computing systems comprehensively. **Application:** The skills acquired in this practical course find direct application in the development of efficient and integrated computing solutions. Students learn to design digital systems, optimize hardware performance, and seamlessly integrate these systems with databases to handle and manipulate data effectively. **Interest:** The Major Computer Science Practical Course is designed to spark interest by offering a hands-on approach to both hardware and software components. Students engage in practical exercises that involve designing digital circuits, implementing database solutions, and integrating these components, fostering a deeper understanding and appreciation for the intricacies of computing systems.

		Connection with Other Courses: This practical course serves as a nexus, connecting various other courses in the computer science curriculum. It lays a foundation for advanced courses in computer organization, embedded systems, software engineering, and database management. The integrated approach ensures students comprehend the synergies between different aspects of computer science. **Demand in the Industry:** Professionals who can seamlessly navigate both digital systems and database management are in high demand. Industries ranging from electronics and telecommunications to software development and data analytics actively seek individuals proficient in both hardware and software aspects, recognizing the practical value of this dual expertise. **Job Prospects:** Graduates from this practical course enjoy promising job prospects in roles that require a holistic understanding of computing systems. Potential job titles include systems architect, database administrator, embedded systems developer, and hardware-software integration specialist. These professionals are well-positioned to contribute to diverse industries seeking comprehensive computing solutions.
2	**Vertical:**	Major
3	**Type:**	Practical
4	**Credits:**	2 credits (1 credit = 30 Hours of Practical work in a semester)
5	**Hours Allotted:**	60 hours
6	**Marks Allotted:**	50 Marks
7	**Course Objectives(CO):** **CO 1.** To verify the truth tables of various logic gates **CO 2.** Develop proficiency in designing and implementing digital circuits. **CO 3.** Explore various components of digital systems, including processors, memory units, and input/output interfaces. **CO 4.** Develop skills in designing and creating relational databases. **CO 5.** Explore the principles of database querying using SQL. **CO 6.** Gain practical knowledge of transaction management and data control in database systems.	

8	**Course Outcomes (OC):** After successful completion of this course, students would be able to - **OC 1.** Verify truth tables of various logic gates **OC 2.** Simplify given Boolean expressions and implement them using Logisim. **OC 3.** Design and validate the operation of various combinational circuits using Logisim. **OC 4.** Understand the behavior and applications of flip-flops in digital systems. **OC 5.** Design and implement expressions using multiplexers/demultiplexers in Logisim. **OC 6.** Create and maintain relational databases, applying normalization principles. **OC 7.** Write simple queries to MySQL related to String, Maths and Date Functions. **OC 8.** Create tables and insert/update/delete data, and query data in a relational DBMS using MySQL commands. **OC 9.** Handle data permissions.
9	**Modules:-** **Module 1 (30 hours):**
	Digital Systems & Architecture – Practical **Logic Gates Truth Table Verification:** Study and verify the truth table of various logic gates (NOT, AND, OR, NAND, NOR, EX-OR, EX-NOR) using Logisim. **Boolean Expression Simplification:** Simplify given Boolean expressions and realize them using Logisim. **Half/Full Adder Design:** Design and verify the operation of a half/full adder using Logisim. **Half/Full Subtractor Design:** Design and verify the operation of a half/full subtractor using Logisim. **4-Bit Magnitude Comparator:** Design a 4-bit magnitude comparator using combinational circuits in Logisim. **Flip-Flop Implementation:** Verify the operation of flip-flops (e.g., D, JK) using logic gates in Logisim. **Counter Operation Verification:** Verify the operation of a counter using Logisim. **4-Bit Shift Register Operation:** Verify the operation of a 4-bit shift register using Logisim. **Multiplexer/Demultiplexer Design:** Design and implement expressions using multiplexers/demultiplexers in Logisim.

	3-Bit Binary Ripple Counter: Design and implement a 3-bit binary ripple counter using JK flip-flops in Logisim. The above practical can be performed using any open source simulator (like Logisim) (Download it from https://sourceforge.net/projects/circuit/)
	Module 2 (30 hours):
	Fundamentals of Database Systems – Practical **Conceptual Design Using ER Diagrams:** Identify entities, attributes, keys, and relationships. Apply generalization and specialization. **Database Management Operations:** View all databases, create a database, view all tables in a database, create tables with and without constraints, perform CRUD operations. **Table Management Operations:** Alter a table, drop/truncate/rename tables, perform backup/restore operations on a database. **Basic Queries and Aggregate Functions:** Execute simple queries and utilize aggregate functions (e.g., COUNT, SUM, AVG). **Advanced Query Functions:** Utilize date, string, and math functions in queries. **Join Queries:** Execute inner and outer join queries. **Subqueries:** Apply subqueries with IN and EXISTS clauses. **ER Model to Relational Model Conversion and Normalization:** Convert ER model to a relational model and apply normalization up to 3rd Normal Form. **Views:** Create views with and without check options, drop views, select data from views. **Data Control Language (DCL) Statements:** Implement DCL statements for granting and revoking permissions. Demonstrate COMMIT and ROLLBACK statements. These experiments can be implemented using a database management system like MySQL.

<table>
<tr><td>10</td><td colspan="2">Text Books
1. R P Jain, Modern Digital Electronics, Tata McGraw Hill Education Pvt. Ltd. , 4th Edition, 2010
2. Murach's MySQL, Joel Murach, 3rd Edition, 3rd Edition, 2019</td></tr>
<tr><td>11</td><td colspan="2">Reference Books
1. MySQL: The Complete Reference, VikramVaswani , McGraw Hill, 2017
2. Learn SQL with MySQL: Retrieve and Manipulate Data Using SQL Commands with Ease, Ashwin Pajankar, BPB Publications, 2020</td></tr>
<tr><td>12</td><td>Internal Continuous Assessment: 40%</td><td>Semester End Examination: 60%</td></tr>
<tr><td>13</td><td>The internal evaluation will be determined by the completion of practical tasks and the submission of corresponding write-ups for each session. Each practical exercise holds a maximum value of 5 marks. The total evaluation, out of 100 marks, should be scaled down to a final score of 20 marks.
Total: 20 marks</td><td>A Semester End Practical Examination of 2 hours duration for 30 marks as per the paper pattern given below.

Certified Journal is compulsory for appearing at the time of Practical Exam
Total: 30 Marks</td></tr>
<tr><td>14</td><td colspan="2">Format of Question Paper:

Total Marks: 30 Duration: 2 Hours
<table>
<tr><th>Question</th><th>Practical Question Based On</th><th>Marks</th></tr>
<tr><td>Q. 1</td><td>Module 1</td><td>12</td></tr>
<tr><td>Q. 2</td><td>Module 2</td><td>12</td></tr>
<tr><td>Q. 3</td><td>Viva</td><td>06</td></tr>
</table></td></tr>
</table>

Vocational & Skill Enhancement Courses (VSEC)

Name of the Course: Introduction to Programming with Python

Sr. No.	Heading	Particulars
1	**Description the course:**	**Introduction:** Introduction to Programming with Python Course serves as an entry point into the world of coding, introducing learners to the versatile and beginner-friendly Python language. Python is renowned for its readability and simplicity, making it an ideal choice for individuals taking their first steps in programming. **Relevance:** In today's digital era, programming skills are increasingly essential across various disciplines. Python, being an interpreted, high-level language, is relevant for diverse applications, from web development and data analysis to artificial intelligence and automation. **Usefulness:** The course provides a foundational understanding of Python syntax, data structures, and control flow, empowering learners to write functional and efficient code. Python's broad applicability makes the skills acquired in this course valuable for numerous programming tasks. **Application:** Upon completion, participants can apply Python to solve real-world problems, automate repetitive tasks, and create simple applications. The practical knowledge gained serves as a stepping stone for more advanced Python courses or specialization in areas like data science or web development. **Interest:** Python's user-friendly syntax and extensive libraries make it an enjoyable language for beginners. The course is designed to spark interest by combining theory with hands-on projects, fostering a passion for coding and problem-solving. **Connection with Other Courses:** Python is a gateway language that seamlessly integrates with other programming languages and technologies. The skills acquired in a Basic Python Programming Course

		provide a solid foundation for advanced programming languages and specialized courses in data science, machine learning, and more. **Demand in the Industry:** Python's popularity in the industry is soaring. Its versatility, readability, and extensive community support have led to its widespread adoption. Professionals proficient in Python are in high demand across various sectors, including technology, finance, healthcare, and academia. **Job Prospects:** Completion of this Course opens doors to entry-level positions in software development, quality assurance, data analysis, and scripting. Python developers are sought after for their ability to quickly prototype solutions and contribute to various stages of software development.
2	**Vertical:**	VSC
3	**Type:**	Practical
4	**Credits:**	2 credits (1 credit = 15 Hours for Theory or 30 Hours of Practical work in a semester)
5	**Hours Allotted:**	60 Hours
6	**Marks Allotted:**	50 Marks
7	**Course Objectives(CO):** **CO 1.** Master Python features, execution, and diverse data types. **CO 2.** Demonstrate expertise in if statements, loops, and control statements. **CO 3.** Efficiently create and manipulate arrays, strings, and data structures. **CO 4.** Apply functions, modules, and strings for versatile programming tasks. **CO 5.** Effectively manage files, utilize regular expressions, and work with date and time.	
8	**Course Outcomes (OC):** **OC 1.** Apply Python features for diverse programming tasks confidently. **OC 2.** Implement control flow statements for precise program execution. **OC 3.** Manipulate arrays, strings, and data structures with precision and ease. **OC 4.** Create modular, efficient code using functions, modules, and strings. **OC 5.** Skillfully manage files, utilize regular expressions, and work with date and time for program efficiency.	
9	**Modules:-** **Module (30 hours):**	
	Overview and Basic Elements of Python Programming: Features of Python, Execution of a Python Program, Flavours of Python, Innards of Python, Python Interpreter, Comments, Docstrings, IDLE, Data types, Dictionary, Sets, Mapping, Basic Elements of Python, Variables, Input Function, Output Statements, Command	

	Line Arguments. Operators, Precedence of Operators, Associativity of Operators **Control Statements:** The if statement, The if … else Statement, The if … elif … else Statement, Loop Statement- while loop, for loop, Infinite loop, Nested loop, The else suite, break statement, continue statement, pass statement, assert statement, return statement. **Arrays:** Creating Arrays, Indexing and Slicing of Arrays, Basic Array Operations, Arrays Processing, Mathematical Operations on Array, Aliasing Arrays, Slicing and Indexing in NumPy Arrays, Basic slicing, Advanced Indexing, Dimensions and Attributes of an Array **Functions:** Function definition and call, Returning Results, Returning Multiple Values from a Function, Built-in Functions, Difference between a Function and a Method, Pass Value by Object Reference, Parameters and Arguments, Recursive Functions, Anonymous or Lambda Functions. Modules in Python. **Strings:** Creating Strings, Functions of Strings, Working with Strings, Formatting Strings, Finding the Number of Characters and Words, Inserting Substrings into a String.
	Module (30 hours):
	Exploring List, Tuples and Dictionaries: Lists, List Functions and Methods, List Operations, List Slices, Nested Lists, Tuples, Functions in Tuple. **Working with Dictionaries:** Creating a Dictionary, Operators in Dictionary, Dictionary Methods, Using for Loop with Dictionaries, Operations on Dictionaries **Files in Python**: Opening and Closing a File, Working with Text Files, , Working with Binary Files, The 'with' statement, Pickle in Python, The seek() and tell() Methods, Random Accessing of Binary Files, Zipping and Unzipping Files, Working with Directories **Regular Expressions:** Introduction, Sequence Characters in Regular Expressions, Special Characters in Regular Expressions, Using Regular Expression on Files, Retrieving Information from an HTML File **Date And Time in Python:** Time, Date, Date and Time Now, combining date and times, formatting date and time, Finding and comparing dates, Sorting dates, Knowing the Time taken by a Program, Working with Calendar Module
10	**Text Books** 1. Practical Programming: An Introduction to Computer Science Using Python 3, Paul Gries , Jennifer Campbell, Jason Montojo, Pragmatic Bookshelf, 2nd Edition, 2014 2. Programming through Python, M. T Savaliya, R. K. Maurya & G M Magar, Sybgen Learning India, 2020
11	**Reference Books** 1. Python: The Complete Reference, Martin C. Brown, McGraw Hill, 2018 2. Beginning Python: From Novice to Professional, Magnus Lie Hetland, Apress, 2017

<table>
<tr><td></td><td colspan="2">3. Programming in Python 3, Mark Summerfield, Pearson Education, 2nd Ed, 2018
4. Python Programming: Using Problem Solving Approach, ReemaThareja, Oxford Univeristy Press, 2017
5. Let Us Python, Yashwant. B. Kanetkar, BPB Publication, 2019</td></tr>
<tr><td>12</td><td>Internal Continuous Assessment: 40%</td><td>Semester End Examination: 60%</td></tr>
<tr><td>13</td><td>The internal evaluation will be determined by the completion of practical tasks and the submission of corresponding write-ups for each session. Each practical exercise holds a maximum value of 5 marks. The total evaluation, out of 50 marks, should be scaled down to a final score of 20 marks.
Total: 20 marks</td><td>A Semester End Practical Examination of 2 hours duration for 30 marks as per the paper pattern given below.

Certified Journal is compulsory for appearing at the time of Practical Exam
Total: 30 Marks</td></tr>
<tr><td>14</td><td colspan="2">Format of Question Paper:

Total Marks: 30 Duration: 2 Hours
<table>
<tr><th>Question</th><th>Practical Question Based On</th><th>Marks</th></tr>
<tr><td>Q. 1</td><td>Module 1</td><td>12</td></tr>
<tr><td>Q. 2</td><td>Module 2</td><td>12</td></tr>
<tr><td>Q. 3</td><td>Viva</td><td>06</td></tr>
</table></td></tr>
</table>

Name of the Course: Statistics with R Programming

Sr. No.	Heading	Particulars
1	**Description the course:**	**Introduction:** This course provides an immersive exploration into the world of statistical computing and data analysis. Developed specifically for statistical computing and graphics, R is an open-source language that has become a standard tool for professionals in various fields. **Relevance:** In the era of big data and analytics, R programming is highly relevant. It is widely used for statistical modeling, data visualization, and machine learning, making it an indispensable skill for individuals in data-centric roles. **Usefulness:** The course equips participants with the ability to manipulate data, perform statistical analyses, and create visualizations. R's versatility makes it valuable for both beginners entering the field and seasoned professionals enhancing their analytical toolkit. **Application:** R programming finds application across diverse domains, including finance, healthcare, marketing, and academia. Participants can apply R to solve real-world problems, extract insights from data, and make informed decisions. **Interest:** The R programming course often sparks interest due to its hands-on nature. Participants engage in practical exercises, exploring datasets, creating visualizations, and developing statistical models, fostering a deep understanding of data analytics. **Connection with Other Courses:** This course forms a symbiotic connection with other data-centric courses. It complements studies in statistics, machine learning, and data science, providing a foundation for advanced analytics. **Demand in the Industry:** Professionals with R programming skills are in high demand. Industries ranging from finance to healthcare seek individuals who can leverage R for data analysis and

		decision-making, contributing to evidence-based practices. **Job Prospects:** Graduates from an R programming course find diverse job prospects. Roles may include data analyst, statistician, business intelligence analyst, and data scientist. These professionals are sought after for their ability to derive actionable insights from data.
2	**Vertical:**	SEC
3	**Type:**	Practical
4	**Credits:**	2 credits (1 credit = 15 Hours for Theory or 30 Hours of Practical work in a semester)
5	**Hours Allotted:**	60 Hours
6	**Marks Allotted:**	50 Marks
7	**Course Objectives(CO):** **CO 1.** Understand R basics, set up R Studio, and customize the environment.. **CO 2.** Master R expressions, assignments, loops, and decision-making. **CO 3.** Develop proficiency in using R data structures: vectors, matrices, lists, and data frames. **CO 4.** Demonstrate expertise in character strings manipulation in R. **CO 5.** Apply built-in statistical functions, regression analysis, and distribution functions fluently.	
8	**Course Outcomes (OC):** **OC 1.** Confidently navigate Studio, R GUI, and manage data in R. **OC 2.** Fluent implementation of expressions, assignments, and loops in R. **OC 3.** Use R data structures for effective data management. **OC 4.** Efficiently manipulate and operate on character strings in R. **OC 5.** Apply statistical functions, regression analysis, and distribution functions with confidence.	
9	**Modules:-** **Module 1 (30 hours):**	
	Exploring R Language and Setting Up environment: Introduction to R, Terminologies in R, R Environment, Installing R, Studio, and R Commander, Customizing Studio, Data Management in Studio, R Graphical User Interface (R GUI), Working with R Scripts **Implementing ting Expression:** Expressions, assignment, Decision making, Loops, data and time options in R **Essential Data Structures in R:** Vectors, Matrix, Arrays, Lists, Data frames, Functions **Implementing Strings in R**: Character strings in R, Character Strings, , Strings and R objects, String Manipulation: Printing Characters, Basic String Manipulations, String Operations	

<table>
<tr><td></td><td colspan="2">Module 2 (30 hours):</td></tr>
<tr><td></td><td colspan="2">Built-in statistical functions in R: mean() function, Median, Standard Deviation, Some other built-in statistical functions,

Regression Analysis: Regression Analysis-Linear Regression and Multiple Regression, Normal Distribution- dnorm(),,pnorm(),qnorm(),rnorm()

Binomial Distribution: dbinom(),pbinom(),qbinom(),rbinom() Functions, Time Series Analysis

Visualizing and analysing Data in R: Tabulation, Contingency Tables, Making R Contingency Tables, Making R Custom Contingency Tables, Selection of Parts in a Table Object, Conversion of an Object into the Table, Testing Table Objects, Making R Complex Tables, Representing data through Cross Tabulation

Graphical Models & analysis: Plots made of Single Plots made of Two Variables , Variable, Plots made of Multiple Variables, Special Plots, Storing Graphics</td></tr>
<tr><td>10</td><td colspan="2">Text Books
1. Statistical Programming in R, K.G. Srinivasa G.M. Siddesh,Chetan Shetty , Oxford University Press, 2017
2. Learning R: A Language for Data Analytics and Visualization, Sybgen Learning, R. K. Maurya, 2021
3. Introduction to Statistics and Data Analysis With Exercises, Solutions and Applications in R: Heumann, Christian, Schomaker, Michael, Shalabh, Publisher” Springer 2016</td></tr>
<tr><td>11</td><td colspan="2">Reference Books
1. Learning R Programming, Kun Ren, Packt Publishing, 2018
2. R Programming for Statistics and Data Science(Video), 365 Careers, Packt, 2018
3. R Programming Fundamentals, Kaelen Medeiros, Oreily-Packt Publishing</td></tr>
<tr><td>12</td><td>Internal Continuous Assessment: 40%</td><td>Semester End Examination: 60%</td></tr>
<tr><td>13</td><td>The internal evaluation will be determined by the completion of practical tasks and the submission of corresponding write-ups for each session. Each practical exercise holds a maximum value of 5 marks. The total evaluation, out of 50 marks, should be scaled down to a final score of 20 marks.

Total: 20 marks</td><td>A Semester End Practical Examination of 2 hours duration for 30 marks as per the paper pattern given below.

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

Total: 30 Marks</td></tr>
<tr><td>14</td><td colspan="2">Format of Question Paper:

Total Marks: 30 Duration: 2 Hours
<table>
<tr><th>Question</th><th>Practical Question Based On</th><th>Marks</th></tr>
<tr><td>Q. 1</td><td>Module 1</td><td>12</td></tr>
<tr><td>Q. 2</td><td>Module 2</td><td>12</td></tr>
<tr><td>Q. 3</td><td>Viva</td><td>06</td></tr>
</table></td></tr>
</table>

Name of the Course: LINUX Operating System

Sr. No.	Heading	Particulars
1	**Description the course:**	**Introduction:** The Linux Operating System course is a foundational exploration into the world of computing, providing students with essential knowledge about this open-source and widely used operating system. **Relevance:** Linux is integral to various industries, from server administration to software development, cybersecurity, cloud computing, and IoT, making the course highly relevant in today's digital landscape. **Usefulness:** Linux dominates global server environments, making it a crucial skill for managing and maintaining servers efficiently. Many development tools and environments are Linux-based, enhancing a developer's capabilities. Linux, well-known for its robust security features, plays a pivotal role in the field of cybersecurity, making Linux knowledge invaluable for professionals in this domain. Popular cloud platforms extensively use Linux, making familiarity with it beneficial for cloud administrators. Linux's prevalence in IoT devices and embedded systems underscores its importance for professionals working in these emerging fields. **Application:** The course introduces students to the core principles and practical applications of Linux, covering areas such as server administration, software development, cybersecurity, cloud computing, and IoT. **Interest:** With its open-source nature and versatile applications, Linux attracts individuals who appreciate efficient command-line tools and those interested in stability, reliability, and the command-line interface. **Connection with Other Courses:** The course seamlessly integrates with network administration courses by incorporating essential Linux commands. It also aligns with various software development courses, fostering a comprehensive

		understanding of computing environments. **Demand in the Industry:** The industry recognizes the stability, security, and cost-effectiveness of Linux, resulting in a consistently high demand for professionals with Linux expertise. **Job Prospects:** Graduates of the Linux Operating System course are well-positioned for diverse roles, including system administrators, network administrators, DevOps engineers, cloud administrators, cybersecurity analysts, and software developers.
2	**Vertical:**	SEC
3	**Type:**	Practical
4	**Credits:**	2 credits (1 credit = 15 Hours for Theory or 30 Hours of Practical work in a semester)
5	**Hours Allotted:**	60 Hours
6	**Marks Allotted:**	50 Marks
7	**Course Objectives(CO):** **CO 1.** To learn basic concepts of Linux in terms of operating system **CO 2.** To learn use of various shell commands with regular expressions **CO 3.** To set Linux Environment variables and learn setting file permissions to maintain Linux security implementation **CO 4.** To learn various editors available in Linux OS and learn shell scripting. **CO 5.** To learn installation of compilers and programming using C and Python languages on Linux platform.	
8	**Course Outcomes (OC):** **OC 1.** Work with Linux file system structure, Linux Environment **OC 2.** Handle shell commands for scripting, with features of regular expressions, redirections **OC 3.** Implement file security permissions **OC 4.** Work with vi, sed and awk editors for shell scripting using various control structures **OC 5.** Install software like compilers and develop programs in C and Python programming languages on Linux Platform	
9	**Modules:-** **Module (30 hours):**	
	Introduction to Linux Operating System and Basics: History of Linux, GNU Info and Utilities, Various Linux Distributions, The Unix/Linux architecture, Features of Unix/Linux **Installation of Ubuntu Linux Operating System:** Booting and Installing from USB/DVD, Using Ubuntu Software Center / Using Synaptic, Exploring useful	

	software packages **Becoming an Ubuntu Power User:** Administering system and user settings, Learning Unity keyboard shortcuts, Using the Terminal **Linux Basics:** Starting the shell, Shell prompt, Command structure, File Systems and Directory Structure, man pages, more documentation pages **File System Commands:** touch, help, man, more, less, pwd, cd, mkdir, rmdir, ls, find, etc. **File Handling Commands:** cat, cp, rm, mv, more, file, wc, od, cmp, diff, comm, gzip, gunzip, zip, unzip, tar, ln, umask, etc. **General Purpose Utility Commands:** cal, date, echo, man, printf, passwd, script, who, uname, tty, stty, etc. **Linux File Permissions:** Understanding Linux file permissions, Using Linux groups. Decoding file permissions, Changing security settings, chmod, chown, chgrp
	Module (30 hours):
	Linux Security: Understanding Linux Security, Uses of root, sudo command, Working with passwords, Understanding ssh **Networking Commands:** who, whoami, ping, telnet, ftp, ssh, etc. **Editors:** vi, sed, awk **Simple Filters and I/O Redirection**: head, tail, cut, paste, sort, grep family, tee, uniq, tr, etc. **Shell Scripting:** Defining variables, reading user input, exit and exit status commands, expr, test, [], if conditional, logical operators, Conditions (for loop, until loop, and while loop), arithmetic operations, Redirecting input/output in scripts, creating your own redirection. **Working and Managing Processes:** sh, ps, kill, nice, at, batch, etc. **Job scheduling commands:** ps, nice, renice, at, batch, cron table Installation of C/C++/Java/Python Compiler and Environment Setup and Basic programming using C and Python languages.
10	**Text Books** 1. Linux Command line and Shell Scripting Bible, Richard Blum, Wiley India. 2. Unix: Concepts and Applications, Sumitabha Das, 4th Edition, McGraw Hill. 3. Official Ubuntu Book, Matthew Helmke& Elizabeth K. Joseph with Jose Antonio Rey and Philips Ballew, 8th Ed.
11	**Reference Books** 1. Linux Administration: A Beginner's Guide, Fifth Edition, Wale Soyinka, Tata McGraw-Hill, 2008. 2. Linux: Complete Reference, Richard Petersen, 6th Edition, Tata McGraw-Hill 3. Beginning Linux Programming, Neil Mathew, 4th Edition, Wiley Publishing, 2008.

<table>
<tr><td>12</td><td>Internal Continuous Assessment: 40%</td><td>Semester End Examination: 60%</td></tr>
<tr><td>13</td><td>The internal evaluation will be determined by the completion of practical tasks and the submission of corresponding write-ups for each session. Each practical exercise holds a maximum value of 5 marks. The total evaluation, out of 50 marks, should be scaled down to a final score of 20 marks.
Total: 20 marks</td><td>A Semester End Practical Examination of 2 hours duration for 30 marks as per the paper pattern given below.

Certified Journal is compulsory for appearing at the time of Practical Exam
Total: 30 Marks</td></tr>
<tr><td>14</td><td colspan="2">Format of Question Paper:

Total Marks: 30 Duration: 2 Hours
<table>
<tr><th>Question</th><th>Practical Question Based On</th><th>Marks</th></tr>
<tr><td>Q. 1</td><td>Module 1</td><td>12</td></tr>
<tr><td>Q. 2</td><td>Module 2</td><td>12</td></tr>
<tr><td>Q. 3</td><td>Viva</td><td>06</td></tr>
</table></td></tr>
</table>

Sem – II

Mandatory Courses

Name of the Course: Design and Analysis of Algorithms

Sr. No.	Heading	Particulars
1	**Description the course:**	**Introduction:** The Design and Analysis of Algorithms course is a fundamental exploration into the systematic study of algorithms, their design principles, and the analysis of their efficiency. It forms the backbone of computer science education, providing essential skills for solving complex computational problems. **Relevance:** In the ever-evolving landscape of computer science, the Design and Analysis of Algorithms course is highly relevant. It equips students with the intellectual tools necessary to address challenges in diverse areas, from software development to artificial intelligence. **Usefulness:** This course is instrumental in cultivating algorithmic thinking. Participants learn to devise efficient algorithms, analyze their correctness, and evaluate their performance, essential skills for creating optimized solutions in various computing applications. **Application:** The knowledge gained from this course finds application in a myriad of scenarios, from developing efficient search and sorting algorithms to optimizing resource utilization in network design and artificial intelligence. **Interest:** The course often captivates students due to its intellectual challenges and problem-solving nature. Participants engage in dissecting complex problems, devising algorithmic solutions, and analyzing their efficiency, fostering a deep appreciation for algorithmic thinking. **Connection with Other Courses:** The Design and Analysis of Algorithms course establishes vital connections with other computer science disciplines. It forms the basis for advanced courses in data structures, algorithmic complexity, and computational theory, providing a holistic understanding of computation.

		Demand in the Industry: Professionals well-versed in algorithm design and analysis are in high demand. Industries ranging from technology and finance to healthcare actively seek individuals who can develop efficient algorithms to solve complex problems and enhance system performance. **Job Prospects:** Graduates from a Design and Analysis of Algorithms course find themselves well-positioned for various roles, including software engineer, algorithm developer, data scientist, and research scientist. These professionals are valued for their ability to devise elegant and efficient solutions to computational challenges.
2	**Vertical:**	Major
3	**Type:**	Theory
4	**Credits:**	2 credits (1 credit = 15 Hours for Theory or 30 Hours of Practical work in a semester)
5	**Hours Allotted:**	30 Hours
6	**Marks Allotted:**	50 Marks
7	**Course Objectives(CO):** **CO 1.** To make students understand the basic principles of algorithm design **CO 2.** To give idea to students about the theoretical background of the basic data structures **CO 3.** To familiarize the students with fundamental problem-solving strategies like searching, sorting, selection, and recursion and help them to evaluate efficiencies of various algorithms. **CO 4.** To teach students the important algorithm design paradigms and how they can be used to solve various real world problems	
8	**Course Outcomes (OC):** **OC 1.** Students should be able to understand and evaluate efficiency of the programs that they write based on performance of the algorithms used. **OC 2.** Students should be able to appreciate the use of various data structures as per need **OC 3.** To select, decide and apply appropriate design principle by understanding the requirements of any real life problems.	
9	**Modules:-** **Module 1 (15 hours):**	
	Introduction to algorithms - What is algorithm, analysis of algorithm, Types of complexity, Running time analysis, How to Compare Algorithms, Rate of Growth, Types of Analysis, Asymptotic Notation, Big-O Notation, Omega-Ω Notation, Theta-Θ Notation, Asymptotic Analysis, Performance characteristics of algorithms,	

	Estimating running time / number of steps of executions on paper, Idea of Computability **Introduction to Data Structures -** What is data structure, types, Introduction to Array(1-d & 2-d), Stack and List data structures, operations on these data structures, advantages disadvantages and applications of these data structures like solving linear equations, Polynomial Representation, Infix-to-Postfix conversion. **Recursion -** What is recursion, Recursion vs Iteration, recursion applications like Factorial of a number, Fibonacci series & their comparative analysis with respect to iterative version, Tower of Hanoi problem. **Basic Sorting Techniques** - Bubble, Selection and Insertion Sort & their comparative analysis
	Module 2 (15 hours):
	Searching Techniques - Linear Search and its types, Binary Search and their comparative analysis, Selection Techniques - Selection by Sorting, Partition-based Selection Algorithm, Finding the Kth Smallest Elements in Sorted Order & their comparative analysis, String Algorithms - Pattern matching in strings, Brute Force Method & their comparative analysis **Algorithm Design Techniques** - Introduction to various types of classifications/design criteria and design techniques, Greedy Technique - Concept, Advantages & Disadvantages, Applications, Implementation using problems like - file merging problem. Divide-n-Conquer - Concept, Advantages & Disadvantages, Applications, Implementation using problems like - merge sort, Strassen's Matrix Multiplication **Dynamic Programming -** Concept, Advantages & Disadvantages, Applications, Implementation using problems like - Fibonacci series, Factorial of a number, Longest Common subsequence **Backtracking Programming -** Concept, Advantages & Disadvantages, Applications, Implementation using problems like N-Queen Problem
10	**Text Books** 1. Data Structure and Algorithm Using Python, Rance D. Necaise, Wiley India Edition, 2016. 2. Data Structures and Algorithms Made Easy, Narasimha Karumanchi, CareerMonk Publications, 2016. 3. Introduction to Algorithms, Thomas H. Cormen, 3rd Edition, PHI.
11	**Reference Books** 1. Introduction to the Design and Analysis of Algorithms, Anany Levitin, Pearson, 3rd Edition, 2011. 2. Design and Analysis of Algorithms, S. Sridhar, Oxford University Press, 2014.

<table>
<tr><td>12</td><td>Internal Continuous Assessment: 40%</td><td>Semester End Examination: 60%</td></tr>
<tr><td>13</td><td>Continuous Evaluation through:
Class Test on Module 1: 10 marks
Class Test on Module 2: 10 marks
Average of 2 Class Tests: 10 marks
Assignment on Module 1: 5 marks
Assignment on Module 2: 5 marks
Total of 2 Assignments: 10 marks
Total: 20 marks</td><td>Evaluation through:
A Semester End Theory Examination of 1 hour duration for 30 marks as per the paper pattern given below.
Total: 30 marks</td></tr>
<tr><td>14</td><td colspan="2">Format of Question Paper:

Total Marks: 30 Duration: 1 Hour
<table>
<tr><th>Question</th><th>Based On</th><th>Options</th><th>Marks</th></tr>
<tr><td>Q. 1</td><td>Module 1</td><td>Any 2 out of 4</td><td>10</td></tr>
<tr><td>Q. 2</td><td>Module 2</td><td>Any 2 out of 4</td><td>10</td></tr>
<tr><td>Q. 3</td><td>Module 1 & 2</td><td>Any 2 out of 4</td><td>10</td></tr>
</table></td></tr>
</table>

Name of the Course: Introduction to OOP using C++

Sr. No.	Heading	Particulars
1	**Description the course:**	**Introduction:** The Introduction to Object-Oriented Programming (OOP) using C++ course is a foundational exploration into the principles of object-oriented programming, using the C++ programming language. This course serves as a gateway for students to understand and apply key concepts in software design and development. **Relevance:** In the contemporary software development landscape, understanding OOP principles is crucial. The C++ language, with its strong support for object-oriented features, is widely used in building robust and efficient software systems. This course is, therefore, highly relevant to the needs of modern programming. **Usefulness:** The course is instrumental in imparting essential programming paradigms such as encapsulation, inheritance, and polymorphism. Participants gain valuable skills in designing modular and reusable code, contributing to the creation of scalable and maintainable software solutions. **Application:** The concepts learned in this course find direct application in software development. Participants learn to structure code using classes and objects, facilitating the creation of efficient and well-organized programs. **Interest:** The course often captivates students due to its practical and creative aspects. Through hands-on projects, participants engage in designing and implementing solutions using OOP principles, fostering a deep interest in software design and development. **Connection with Other Courses:** This course establishes strong connections with other programming and software engineering courses. It lays the groundwork for advanced studies in software architecture, design patterns, and application development, providing a seamless transition to more

		complex programming concepts. **Demand in the Industry:** Professionals with a solid understanding of OOP using C++ are in high demand. Industries ranging from software development to embedded systems actively seek individuals who can leverage OOP principles to create efficient, modular, and maintainable code. **Job Prospects:** Students completing this course may find diverse job prospects. Roles may include software developer, systems analyst, application architect, and embedded systems engineer. These professionals are valued for their ability to contribute to the creation of robust and scalable software solutions.
2	**Vertical:**	Major
3	**Type:**	Theory
4	**Credits:**	2 credits (1 credit = 15 Hours for Theory or 30 Hours of Practical work in a semester)
5	**Hours Allotted:**	30 Hours
6	**Marks Allotted:**	50 Marks
7	**Course Objectives(CO):** **CO 1.** To make learner understand the concepts of OOP **CO 2.** To make learner understand the design of OOP through UML **CO 3.** To make learner familiar with the syntax of C++ **CO 4.** To make learner Analyze and implement concepts of OOP **CO 5.** To make learner create programs relating to OOP concepts	
8	**Course Outcomes (OC):** **OC 1.** The learner will be able to understand, remember, demonstrate, explain and describe concept of OOP **OC 2.** The learner will be able to design UML based diagrams **OC 3.** The learner will be able to illustrate the different types of control statements in C++ **OC 4.** The learner will be able to analyze and implement concept of OOP **OC 5.** The learner will be able to write and create programs relating to OOP concepts	
9	**Modules:-** **Module 1 (15 hours):**	
	Introduction to Programming Concepts: Object oriented programming paradigm, basic concepts of object oriented programming, benefits of object oriented programming, object oriented languages, applications of object oriented programming.Tokens-keywords, identifiers, constants-integer, real, character and string constants, backslash constants, features of C++ and its basic structure, simple	

	C++ program without class, compiling and running C++ program. **Data Types, Data Input Output and Operators:** Basic data types,variables, rules for naming variables, programming constants, the type cast operator, implicit and explicit type casting, cout and cin statements,operators, precedence of operators. **Decision Making, Loops, Arrays and Strings:** Conditional statements-if,if...else, switch loops- while, do...while, for, types of arrays and string and string manipulations **Unified Modeling Language (UML):** Introduction to UML & class diagrams. **Classes, Abstraction & Encapsulation:** Classes and objects, Dot Operator, data members, member functions, passing data to functions, scope and visibility of variables in function. **Constructors and Destructors:** Default constructor, parameterized constructor, copy constructor, private constructor, destructors. **Working with objects:** Accessor - mutator methods, static data and static function, access specifiers, array of objects.
	Module 2 (15 hours):
	Polymorphism - Binding-static binding & overloading, constructor overloading function overloading, operator overloading, overloading unary and binary operators. **Modelling Relationships in Class Diagrams:** Association, Aggregation-Composition and examples covering these principles **Inheritance:** Defining base class and its derived class, access specifiers, types of inheritance-single, multiple, hierarchical, multilevel, hybrid inheritance, friend function and friend class, constructors in derived classes. **Modelling Relationships:** Generalization-Specialization and examples covering these principles **Run time Polymorphism -** Dynamic Binding, Function overriding, virtual function, pure virtual function, virtual base class, abstract class. **Pointers:** Introduction to pointers, * and & operators, assigning addresses to pointer variables, accessing values using pointers, pointers to objects & this pointer, pointers to derived classes **File Handling:** File Stream classes, opening and closing file-file opening modes, text file handling, binary file handling. **Applying OOP to solve real life applications:** To cover case studies like library management, order management etc. to design classes covering all relationships
10	**Text Books** 1. Object Oriented Programming with C++, Balagurusamy E., 8th Edition, McGraw Hill Education India. 2. UML & C++: A Practical Guide to Object Oriented Development, Lee/Tepfenhart, Pearson Education, 2nd Edition2015

<table>
<tr><td>11</td><td colspan="2">Reference Books
1. Mastering C++ by Venugopal, Publisher: McGraw-Hill Education, 2017
2. Let Us C++ by KanetkarYashwant, Publisher: BPB Publications, 2020
3. Object Oriented Analysis and Design by Timothy Budd TMH, 2001</td></tr>
<tr><td>12</td><td>Internal Continuous Assessment: 40%</td><td>Semester End Examination: 60%</td></tr>
<tr><td>13</td><td>Continuous Evaluation through:
Class Test on Module 1: 10 marks
Class Test on Module 2: 10 marks
Average of 2 Class Tests: 10 marks
Assignment on Module 1: 5 marks
Assignment on Module 2: 5 marks
Total of 2 Assignments: 10 marks
Total: 20 marks</td><td>Evaluation through:
A Semester End Theory Examination of 1 hour duration for 30 marks as per the paper pattern given below.
Total: 30 marks</td></tr>
<tr><td>14</td><td colspan="2">Format of Question Paper:

Total Marks: 30 Duration: 1 Hour
<table>
<tr><th>Question</th><th>Based On</th><th>Options</th><th>Marks</th></tr>
<tr><td>Q. 1</td><td>Module 1</td><td>Any 2 out of 4</td><td>10</td></tr>
<tr><td>Q. 2</td><td>Module 2</td><td>Any 2 out of 4</td><td>10</td></tr>
<tr><td>Q. 3</td><td>Module 1 & 2</td><td>Any 2 out of 4</td><td>10</td></tr>
</table></td></tr>
</table>

Name of the Course: Computer Science Practical 2

Sr. No.	Heading	Particulars
1	**Description the course:**	**Introduction:** The Computer Science Practical Course covering Design and Analysis of Algorithms and Object-Oriented Programming (OOP) using C++ is a comprehensive exploration into fundamental computer science concepts and practical programming skills. It integrates the study of algorithmic design with hands-on application using the C++ programming language. **Relevance:** In the dynamic field of computer science, the integration of algorithmic design and object-oriented programming is highly relevant. This course equips students with essential skills to solve complex problems, design efficient algorithms, and implement practical solutions using the OOP paradigm in C++. **Usefulness:** The course is invaluable for developing a strong foundation in algorithmic thinking and software design. Students learn to analyze algorithm efficiency, apply OOP principles for code modularity, and create robust software solutions, enhancing their overall programming proficiency. **Application:** The concepts acquired in this practical course find direct application in real-world scenarios. Students engage in hands-on projects where they design and implement algorithms, analyze their performance, and develop software applications using object-oriented principles in C++. **Interest:** The practical nature of the course often captivates students. Through project-based learning, participants apply algorithmic strategies, design class hierarchies, and implement solutions in C++, fostering a deep interest in problem-solving and software development. **Connection with Other Courses:** This practical course establishes a strong connection with other computer science courses. It lays the groundwork for advanced studies in algorithmic complexity, data structures, software engineering, and advanced topics in object-oriented programming, providing a well-rounded education.

		Demand in the Industry: Professionals with proficiency in algorithmic design and object-oriented programming in C++ are in high demand. Industries spanning software development, technology, and finance actively seek individuals who can apply these skills to create efficient and scalable software solutions. **Job Prospects:** Graduates from this practical course have diverse job prospects. Roles may include software engineer, algorithm developer, systems analyst, or application developer. These professionals are valued for their ability to contribute to algorithmically optimized, modular, and maintainable software.
2	**Vertical:**	Major
3	**Type:**	Practical
4	**Credits:**	2 credits (1 credit = 30 Hours of Practical work in a semester)
5	**Hours Allotted:**	60 Hours
6	**Marks Allotted:**	50 Marks
7	**Course Objectives(CO):** **CO 1.** Analyze and implement algorithms for common computational problems. **CO 2.** Implement algorithms using divide and conquer strategies. **CO 3.** Apply dynamic programming techniques to solve optimization problems. **CO 4.** Implement and analyze algorithms based on greedy strategies. **CO 5.** Comprehend the principles of object-oriented programming. **CO 6.** Design and implement classes and objects in C++. **CO 7.** Implement single, multiple, and hierarchical inheritance. **CO 8.** Implement operator overloading for user-defined types. **CO 9.** Understand the impact of access specifiers on class members.	
8	**Course Outcomes (OC):** **OC 1.** Design and implement algorithms for various problem domains. **OC 2.** Evaluate and compare the time and space complexities of algorithms. **OC 3.** Apply divide and conquer strategies to solve computational problems. **OC 4.** Utilize dynamic programming techniques for optimization problems. **OC 5.** Implement and analyze algorithms based on greedy strategies. **OC 6.** Design and implement classes and objects in C++. **OC 7.** Apply inheritance and polymorphism concepts in program development. **OC 8.** Implement operator overloading for enhanced class functionality. **OC 9.** Utilize advanced features like friend functions, inline functions, and this pointer. **OC 10.** Understand the impact of scope specifiers on class members.	

9	**Modules:-** **Module 1 (30 hours):**
	Design & Analysis of Algorithms – Practical **Array Operations:** Implement programs for 1-d arrays, Implement programs for 2-d arrays. **List-Based Stack Operations:** Create a list-based stack and perform stack operations. **Linear and Binary Search:** Implement linear and binary search algorithms on a list. **Sorting Algorithms:** Implement sorting algorithms (e.g., bubble, selection, insertion). **Nth Max/Min Element:** Implement algorithms to find Nth Max/Min element in a list. **String Pattern Matching:** Implement algorithms to find a pattern in a given string. **Recursion:** Implement recursive algorithms (e.g., factorial, Fibonacci, Tower of Hanoi). **Greedy Algorithm:** Solve problems like file merging and coin change using the Greedy Algorithm. **Divide and Conquer:** Implement algorithms like merge sort and Strassen's Matrix Multiplication. **Dynamic Programming:** Implement algorithms for Fibonacci series and Longest Common Subsequence using dynamic programming.
	Module 2 (30 hours):
	OOPs using C++ – Practical **Introduction to Classes:** Create a simple class with data members and member functions. Demonstrate the use of class instances to access data and invoke member functions. **Branching and Looping with Classes:** Implement programs utilizing branching and looping statements within class methods. **Arrays and Classes:**

	Develop a program that employs one and two-dimensional arrays within a class. Illustrate how classes can handle array-based data structures. **Scope Resolution Operator:** Use the scope resolution operator to declare variables at different scope levels. Display and compare the values of variables with different scopes. **Constructors and Destructors:** Implement programs showcasing various types of constructors and destructors. Explore default, parameterized, copy constructors, and destructor functionalities. **Access Specifiers:** Demonstrate the use of public, protected, and private scope specifiers within a class. Understand the impact of different access specifiers on class members. **Inheritance:** Implement classes to demonstrate single and multilevel inheritance scenarios. Showcase how derived classes inherit properties from the base class. Develop programs illustrating multiple and hierarchical inheritance. Create programs that demonstrate the interaction between inheritance and derived class constructors. Understand the order of constructor invocation in the inheritance hierarchy. **Advanced Concepts:** Implement programs showcasing friend functions, inline functions, and the use of the this pointer within classes. **Function Overloading and Overriding:** Develop programs to demonstrate function overloading and overriding within classes. **Pointers and File Handling:** Explore the use of pointers within classes, emphasizing dynamic memory allocation. Develop programs for both text and binary file handling within a class context.
10	**Text Books** 1. Data Structure and Algorithm Using Python, Rance D. Necaise, Wiley India Edition, 2016. 2. Object Oriented Programming with C++, Balagurusamy E., 8th Edition, McGraw Hill Education India.

<table>
<tr><td>11</td><td colspan="2">Reference Books
1. Data Structures and Algorithms Made Easy, Narasimha Karumanchi, CareerMonk Publications, 2016.
2. Let Us C++ by KanetkarYashwant, Publisher: BPB Publications, 2020</td></tr>
<tr><td>12</td><td>Internal Continuous Assessment: 40%</td><td>Semester End Examination: 60%</td></tr>
<tr><td>13</td><td>The internal evaluation will be determined by the completion of practical tasks and the submission of corresponding write-ups for each session. Each practical exercise holds a maximum value of 5 marks. The total evaluation, out of 100 marks, should be scaled down to a final score of 20 marks.

Total: 20 marks</td><td>A Semester End Practical Examination of 2 hours duration for 30 marks as per the paper pattern given below.

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

Total: 30 Marks</td></tr>
<tr><td>14</td><td colspan="2">Format of Question Paper:

Total Marks: 30 Duration: 2 Hours
<table>
<tr><th>Question</th><th>Practical Question Based On</th><th>Marks</th></tr>
<tr><td>Q. 1</td><td>Module 1</td><td>12</td></tr>
<tr><td>Q. 2</td><td>Module 2</td><td>12</td></tr>
<tr><td>Q. 3</td><td>Viva</td><td>06</td></tr>
</table></td></tr>
</table>

Vocational & Skill Enhancement Courses (VSEC)

Name of the Course: Web Designing

Sr. No.	Heading	Particulars
1	**Description the course:**	**Introduction:** The Web Designing Course is an immersive exploration into the core technologies that drive the visual and interactive aspects of the web. Covering HTML, CSS, Javascript, XML, and PHP, this course equips individuals with the skills needed to create dynamic and aesthetically pleasing websites. **Relevance:** In the digital age, web design is paramount. The course remains highly relevant as it introduces participants to the fundamental languages and technologies that form the backbone of modern web development. **Usefulness:** The course is invaluable for anyone interested in creating responsive, user-friendly, and visually appealing websites. Participants gain practical skills in structuring web content, styling layouts, and implementing interactive features. **Application:** The concepts learned in this course find direct application in real-world web development projects. Participants design and build websites, applying HTML for structure, CSS for styling, Javascript for interactivity, XML for data representation, and PHP for server-side scripting. **Interest:** The creative and hands-on nature of web design often captivates students. Through practical exercises, participants engage in designing and developing websites, fostering a deep interest in creating visually engaging online experiences. **Connection with Other Courses:** This course establishes strong connections with various other courses in the field of web development and computer science. It provides a foundation for advanced studies in full-stack development, database management, and server-side scripting.

		Demand in the Industry: Professionals with strong web designing skills are in high demand. Industries spanning e-commerce, technology, and media actively seek individuals who can create user-friendly and visually appealing websites to enhance online presence and user engagement. **Job Prospects:** Graduates from a Web Designing Course find diverse job prospects. Roles may include web designer, front-end developer, UI/UX designer, and web content manager. These professionals are sought after for their ability to create visually stunning and functional web interfaces.
2	**Vertical:**	VSC
3	**Type:**	Practical
4	**Credits:**	2 credits (1 credit = 15 Hours for Theory or 30 Hours of Practical work in a semester)
5	**Hours Allotted:**	60 Hours
6	**Marks Allotted:**	50 Marks
7	**Course Objectives(CO):** **CO 1.** To understand the concept of Web Technologies **CO 2.** To understand the concepts of Hyper Text Markup Language and Cascading Style Sheets. **CO 3.** To learn JavaScript for creating dynamic websites. **CO 4.** To learn various operations performed on data among web applications using XML **CO 5.** To learn Server-Side Programming using PHP	
8	**Course Outcomes (OC):** **OC 1.** Design valid, well-formed, scalable, and meaningful pages using emerging technologies. **OC 2.** Understand the various platforms, devices, display resolutions, viewports, and browsers that render websites **OC 3.** Develop and implement client-side and server-side scripting language programs. **OC 4.** Develop and implement Database Driven Websites. **OC 5.** Design and apply XML to create a markup language for data and document centric applications.	
9	**Modules:-** **Module 1 (30 hours):**	
	HTML: Fundamental Elements of HTML, Formatting Text in HTML, Organizing Text in HTML, List Tags, Links and URLs in HTML, Tables in HTML, Images on a Web Page, Image Formats, Image Maps, Colors, Navigation across multiple pages, Forms in HTML, Interactive Elements, Working with Multimedia - Audio and Video	

	File Formats, HTML elements for inserting Audio / Video on a web page **CSS:** Understanding the Syntax of CSS, CSS Selectors, Inserting CSS in an HTML Document, CSS properties to work with background of a Page, CSS properties to work with Fonts and Text Styles, CSS properties for positioning an element. **JavaScript:** Using JavaScript in an HTML Document, Programming, Fundamentals of JavaScript – Variables, Operators, Control Flow Statements, Popup Boxes, Functions – Defining and Invoking a Function, Defining Function arguments, defining a return Statement, Calling Functions with Timer, JavaScript Objects - String, RegExp, Math, Date, Browser Objects - Window, Navigator, History, Location, Document, Cookies, Document Object Model, Form Validation using JavaScript	
	Module 2 (30 hours):	
	XML: Comparing XML with HTML, Advantages and Disadvantages of XML, Structure of an XML Document, XML Entity References, with Internal / External DTD, XSLT Elements and Attributes **AJAX:** AJAX Web Application Model, How AJAX Works, XMLHttpRequest Object – Properties and Methods, Handling asynchronous requests using AJAX e.g. Mouseover, button click, **PHP:** Variables and Operators, Retrieving data from HTML forms, Program Flow, Arrays, working with Files and Directories, working with Databases, Working with Cookies, Sessions, and Headers	
10	**Text Books** 1. HTML 5 Black Book, Covers CSS 3, JavaScript, XML, XHTML, AJAX, PHP and jQuery, 2ed, Dreamtech Press, 2016 2. Web Programming and Interactive Technologies, scriptDemics, StarEdu Solutions India, 2018 3. PHP: A Beginners Guide, Vikram Vaswani, TMH	
11	**Reference Books** 1. HTML, XHTML, and CSS Bible Fifth Edition, Steven M. Schafer, WILEY, 2011 2. Learning PHP, MySQL, JavaScript, CSS & HTML5, Robin Nixon, O'Reilly, 2018 3. PHP, MySQL, JavaScript & HTML5 All-in-one for Dummies, Steve Suehring, Janet Valade Wiley, 2018	
12	**Internal Continuous Assessment: 40%**	**Semester End Examination: 60%**
13	The internal evaluation will be determined by the completion of practical tasks and the submission of corresponding write-ups for each session. Each practical exercise holds a maximum value of 5 marks. The total evaluation, out of 50 marks, should be scaled down to a final score of 20 marks.	A **Semester End Practical Examination** of **2 hours duration** for **30 marks** as per the paper pattern given below. **Certified Journal** is **compulsory** for appearing at the time of Practical Exam

<table>
<tr><td></td><td>Total: 20 marks</td><td>Total: 30 Marks</td></tr>
<tr><td>14</td><td colspan="2">Format of Question Paper:

Total Marks: 30 Duration: 2 Hours

<table>
<tr><th>Question</th><th>Practical Question Based On</th><th>Marks</th></tr>
<tr><td>Q. 1</td><td>Module 1</td><td>12</td></tr>
<tr><td>Q. 2</td><td>Module 2</td><td>12</td></tr>
<tr><td>Q. 3</td><td>Viva</td><td>06</td></tr>
</table>
</td></tr>
</table>

Name of the Course: Database Management Systems Using PL/SQL

Sr. No.	Heading	Particulars
1	**Description the course:**	**Introduction:** The Database Management Systems (DBMS) Using PL/SQL course is a comprehensive exploration into the principles and practices of managing databases using the powerful PL/SQL language. This course provides participants with the skills needed to design, implement, and maintain robust database systems. **Relevance:** In the era of information technology, databases serve as the backbone of applications. The course is highly relevant as it delves into PL/SQL, a procedural language designed for seamless interaction with Oracle databases, one of the most widely used database management systems. **Usefulness:** The course is invaluable for individuals seeking proficiency in database management. Participants learn to harness the capabilities of PL/SQL for efficient data storage, retrieval, and manipulation, enhancing the functionality and performance of database systems. **Application:** The concepts learned in this course find direct application in real-world scenarios. Participants design and implement database structures, write PL/SQL scripts for data manipulation, and optimize database performance, ensuring the efficient operation of data-centric applications. **Interest:** The hands-on and problem-solving nature of working with databases and PL/SQL often captivates students. Through practical exercises, participants engage in creating and managing databases, fostering a deep interest in efficient data storage and retrieval. **Connection with Other Courses:** This course establishes strong connections with other courses in the field of database management, data analytics, and software development. It provides a foundation for advanced studies in database optimization,

		data warehousing, and application development. **Demand in the Industry:** Professionals proficient in database management using PL/SQL are in high demand. Industries spanning finance, healthcare, and e-commerce actively seek individuals who can design and manage databases to ensure data integrity, security, and optimal performance. **Job Prospects:** Graduates from a DBMS Using PL/SQL course find diverse job prospects. Roles may include database administrator, SQL developer, data analyst, and database architect. These professionals are valued for their ability to create and manage databases critical to organizational success.
2	**Vertical:**	SEC
3	**Type:**	Practical
4	**Credits:**	2 credits (1 credit = 15 Hours for Theory or 30 Hours of Practical work in a semester)
5	**Hours Allotted:**	60 Hours
6	**Marks Allotted:**	50 Marks
7	**Course Objectives(CO):** **CO 1.** To develop understanding of concepts and techniques for data management **CO 2.** To learn about widely used systems for implementation and usage **CO 3.** To develop understanding of Transaction management and crash recovery.	
8	**Course Outcomes (OC):** **OC 1.** Master concepts of stored procedure, functions, cursors and triggers and its use. **OC 2.** Learn about using PL/SQL for data management. **OC 3.** Use efficiently Collections and records. **OC 4.** Understand concepts and implementations of transaction management and crash recovery.	
9	**Modules:-** **Module 1 (30 hours):**	
	Overview of PL/SQL: Advantages of PL/SQL, Main Features of PL/SQL, Architecture of PL/SQL **Fundamentals of PL/SQL:** Character Sets, Lexical Units, Declarations, References to Identifiers, Scope and Visibility of Identifiers, Assigning Values to Variables, Expressions, Error-Reporting Functions, Data Types., Control Statements: Conditional Selection Statements, LOOP Statements, Sequential Control Statements, GOTO, and NULL Statements.	

	Sequences: creating sequences, referencing, altering, and dropping a sequence. **Stored Procedures and Functions:** Procedures: Types and benefits of stored procedures, creating stored procedures, executing stored procedures, altering stored procedures, viewing stored procedures. Functions: Calling function and recursion function. **Collections and Records:** Associative Arrays, Varrays (Variable-Size Arrays), Nested Tables, Collection Constructors, Assigning Values to Collection Variables, Multidimensional Collections, Collection Comparisons, Collection Methods, Collection Types Defined in Package Specifications, Record Variables, Assigning Values to Record Variables. **Error Handling:** Compile-Time Warnings, Overview of Exception Handling, Internally Defined Exceptions, Predefined Exceptions, User- Defined Exceptions, Redeclared Predefined Exceptions, Raising Exceptions Explicitly, Exception Propagation, Unhandled Exceptions.
	Module 2 (30 hours):
	Cursors: Overview of Cursor, Types of cursors, Invalid cursor Exception. **Static SQL:** Description of Static SQL, Cursors Overview, Processing Query Result Sets, Cursor Variables, CURSOR Expressions, **Transaction Processing and Control**: Autonomous Transactions, Commit Protocol, Concurrency Control, Lock Management, Read-Write Locks, Deadlocks Handling, **Dynamic SQL:** Native Dynamic SQL, DBMS_SQL Package, SQL Injection. **Triggers:** Overview of Triggers, implementing triggers – creating triggers, Insert, delete, and update triggers, nested triggers, viewing, deleting, and modifying triggers, enforcing data integrity through triggers. **Packages:** Overview of a Package. Need of Packages, Package Specification, Package Body, Package Instantiation, and Initialization. Create nested tables and work with nested tables.
10	**Text Books** 1. Mastering PL/SQL Through Illustrations: From Learning Fundamentals to Developing Efficient PL/SQL Blocks, Dr. B. Chandra, BPB Publication, 2020 2. Oracle Pl/SQL Training Guide., Training guide, BPB Publications, 2016 3. Raghu Ramakrishnam, Gehrke, Database Management Systems, McGraw-Hill,3rd Edition, 2014 4. Abraham Silberschatz, Henry F. Korth,S. Sudarshan , Database System Concepts, 6th Edition 2019
11	**Reference Books** 1. Ivan Bayross, SQL, PL/SQL -The Programming language of Oracle, B.P.B. Publications 2009 2. Ramez Elmasri & Shamkant B.Navathe, Fundamentals of Database Systems, Pearson Education, 2008

<table>
<tr><td>12</td><td>Internal Continuous Assessment: 40%</td><td>Semester End Examination: 60%</td></tr>
<tr><td>13</td><td>The internal evaluation will be determined by the completion of practical tasks and the submission of corresponding write-ups for each session. Each practical exercise holds a maximum value of 5 marks. The total evaluation, out of 50 marks, should be scaled down to a final score of 20 marks.

Total: 20 marks</td><td>A Semester End Practical Examination of 2 hours duration for 30 marks as per the paper pattern given below.

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

Total: 30 Marks</td></tr>
<tr><td>14</td><td colspan="2">Format of Question Paper:

Total Marks: 30 Duration: 2 Hours

<table>
<tr><th>Question</th><th>Practical Question Based On</th><th>Marks</th></tr>
<tr><td>Q. 1</td><td>Module 1</td><td>12</td></tr>
<tr><td>Q. 2</td><td>Module 2</td><td>12</td></tr>
<tr><td>Q. 3</td><td>Viva</td><td>06</td></tr>
</table>
</td></tr>
</table>

Name of the Course: Advanced Python Programming

Sr. No.	Heading	Particulars
1	**Description the course:**	**Introduction:** The Advanced Python Programming Course is designed to elevate coding skills to a more sophisticated level, offering participants a deeper understanding of Python's advanced features and capabilities. Building upon the foundations laid in basic Python courses, this advanced course delves into complex programming concepts and techniques. **Relevance:** As technology advances, the relevance of Python continues to grow. The Advanced Python Programming Course is a response to the increasing demand for skilled Python developers who can tackle intricate challenges in various domains, including software development, data science, artificial intelligence, and more. **Usefulness:** This course goes beyond basic syntax and introduces participants to advanced Python topics such as decorators, generators, metaclasses, and asynchronous programming. Learners gain valuable insights into optimizing code performance, enhancing code readability, and solving complex problems efficiently. **Application:** Graduates of this course can apply their advanced Python skills to tackle more complex programming tasks, develop scalable applications, and contribute to large-scale software projects. The course's emphasis on practical applications ensures that participants are well-equipped for real-world programming challenges. **Interest:** The course maintains an engaging learning experience, balancing theoretical concepts with hands-on projects that challenge participants to apply their knowledge creatively. This approach fosters a continued interest in Python programming and encourages learners to explore advanced topics with enthusiasm. **Connection with Other Courses:** The knowledge gained in the Advanced Python

		Programming Course establishes a strong foundation for further specialization in advanced Python libraries, frameworks, and application domains. This course acts as a bridge to more specialized fields such as machine learning, web development, and data engineering. **Demand in the Industry:** Professionals with advanced Python skills are highly sought after in the industry. The ability to leverage Python's advanced features for efficient problem-solving, code optimization, and system architecture places graduates of this course in high demand across diverse sectors. **Job Prospects:** Completing the Advanced Python Programming Course opens doors to advanced positions in software development, data engineering, scientific computing, and research. Job prospects include roles such as Python developer, data scientist, machine learning engineer, and backend developer, among others.
2	**Vertical:**	SEC
3	**Type:**	Practical
4	**Credits:**	2 credits (1 credit = 15 Hours for Theory or 30 Hours of Practical work in a semester)
5	**Hours Allotted:**	60 Hours
6	**Marks Allotted:**	50 Marks
7	**Course Objectives(CO):** **CO 1.** Master OOPs principles, solving real-world problems. **CO 2.** Create robust Python classes, transfer members efficiently. **CO 3.** Understand and implement inheritance, utilize advanced polymorphism.. **CO 4.** Implement abstract classes, leverage interfaces for flexible code. **CO 5.** Create and synchronize threads, mitigate deadlock issues.	
8	**Course Outcomes (OC):** **OC 1.** Demonstrate comprehensive OOPs proficiency, apply principles effectively. **OC 2.** Develop efficient, reusable classes, successfully transfer members. **OC 3.** Ability to implement inheritance and apply advanced polymorphism. **OC 4.** Ability to implement abstract classes, demonstrate flexibility through interfaces. **OC 5.** Ability to thread creation, synchronization, and effective deadlock resolution.	

9	**Modules:-** **Module 1 (30 hours):**
	OOPs In Python: Introduction to OOPs, Problems in Procedure Oriented Approach, Features of Object Oriented Programming System (OOPS), Constructors and Destructors, **Classes and Objects-** Creating a Class, Self-Variable, Types of Variables, Types of Methods, Passing Members of One Class to Another Class **Inheritance and Polymorphism:** Types of Inheritance, Constructors in Inheritance, Overriding Super Class Constructors and Methods, super() method, Polymorphism, Duck Typing , Operator Overloading, Method Overloading, Method Overriding **Abstract Classes and Interfaces:** Abstract Class, Abstract Method, Interfaces in Python **Threads in Python:** Creating Threads in Python, Single Tasking and Multitasking, Thread Synchronisation, Deadlock in Threads
	Module 2 (30 hours):
	Working with Databases: DBMS, working with MySQL Database-retrieving, inserting, deleting, updating rows from table, Creating Database Tables through Python **Exceptions:** Errors in a Python Program, Exceptions and Exceptions handling, User Defined Exceptions, Logging Exceptions, **Networking:** TCP/IP Protocol Architecture, , User Datagram Protocol (UDP), FTP Architecture, Web Page Operations, Sending a Simple Mail **Graphical User Interface:** Creating a GUI in Python, Widget classes, Layout Manager, Event Handling **Data Science Tools:** Introduction to NumPy, Matplotlib, pandas, Scipy,
10	**Text Books** 1. Practical Programming: An Introduction to Computer Science Using Python 3, Paul Gries , Jennifer Campbell, Jason Montojo, Pragmatic Bookshelf, 2nd Edition, 2014 2. Programming through Python, M. T Savaliya, R. K. Maurya& G M Magar, Sybgen Learning India, 2020
11	**Reference Books** 1. Python: The Complete Reference, Martin C. Brown, McGraw Hill, 2018 2. Beginning Python: From Novice to Professional, Magnus Lie Hetland, Apress, 2017 3. Programming in Python 3, Mark Summerfield, Pearson Education, 2nd Ed, 2018

<table>
<tr><td>12</td><td>Internal Continuous Assessment: 40%</td><td>Semester End Examination: 60%</td></tr>
<tr><td>13</td><td>The internal evaluation will be determined by the completion of practical tasks and the submission of corresponding write-ups for each session. Each practical exercise holds a maximum value of 5 marks. The total evaluation, out of 50 marks, should be scaled down to a final score of 20 marks.
Total: 20 marks</td><td>A Semester End Practical Examination of 2 hours duration for 30 marks as per the paper pattern given below.

Certified Journal is compulsory for appearing at the time of Practical Exam
Total: 30 Marks</td></tr>
<tr><td>14</td><td colspan="2">Format of Question Paper:

Total Marks: 30 Duration: 2 Hours

<table>
<tr><th>Question</th><th>Practical Question Based On</th><th>Marks</th></tr>
<tr><td>Q. 1</td><td>Module 1</td><td>12</td></tr>
<tr><td>Q. 2</td><td>Module 2</td><td>12</td></tr>
<tr><td>Q. 3</td><td>Viva</td><td>06</td></tr>
</table>
</td></tr>
</table>

Letter Grades and Grade Points:

Semester GPA/ Programme CGPA Semester/ Programme	% of Marks	Alpha-Sign/ Letter Grade Result	Grading Point
9.00 - 10.00	90.0 – 100	O (Outstanding)	10
8.00 - < 9.00	80.0 - < 90.0	A+ (Excellent)	9
7.00 - < 8.00	70.0 - < 80.0	A (Very Good)	8
6.00 - < 7.00	60.0 - < 70.0	B+ (Good)	7
5.50 - < 6.00	55.0 - < 60.0	B (Above Average)	6
5.00 - < 5.50	50.0 - < 55.0	C (Average)	5
4.00 - < 5.00	40.0 - < 50.0	P (Pass)	4
Below 4.00	Below 40.0	F (Fail)	0
Ab (Absent)	----	Ab (Absent)	0

Appendix B

Justification for B.Sc. (Computer Science)

1.	Necessity for starting the course:	The B.Sc. (Computer Science) course is strategically designed to meet the rising demand for skilled professionals while emphasizing innovation. In today's tech-driven era, it addresses the need for individuals proficient in computer science principles, programming, and creative problem-solving. This program not only fills the industry demand for qualified graduates but also instills an innovative mindset, preparing students to drive advancements and address real-world challenges.
2.	Whether the UGC has recommended the course:	Yes
3.	Whether all the courses have commenced from the academic year 2023-24	All courses under the B.Sc. (Computer Science) program have commenced as of the academic year 2023-24. Furthermore, the course has been restructured in alignment with the National Education Policy (NEP) 2020, effective from the academic year 2024-2025.
4.	The courses started by the University are self-financed, whether adequate number of eligible permanent faculties are available?	The courses initiated by the University are self-financed, adhering to the sanction provided by the University of Mumbai to affiliated colleges. The availability of an adequate number of eligible permanent faculties aligns with the self-financed nature of these courses.
5.	To give details regarding the duration of the Course and is it possible to compress the course?	The course duration is three years, spanning six semesters. It is not feasible to compress the course, as the curriculum is structured to ensure comprehensive coverage of the required subjects and allow for effective learning and skill development.
6.	The intake capacity of each course and no. of admissions given in the current academic year:	The intake capacity of the course is 60 students per division. The intake capacity varies across affiliated colleges depending upon the sanction received from the University from time to time.

7.	Opportunities of Employability / Employment available after undertaking these courses:	Upon completion of the B.Sc. (Computer Science) course, students will be well-equipped to pursue various opportunities in the dynamic IT industry, with a strong emphasis on innovation. Graduates will possess the skills required for roles in cutting-edge areas such as software development, data analysis, artificial intelligence, cybersecurity, and more. The curriculum is meticulously designed to align with industry needs and foster a spirit of innovation, making graduates not only highly sought after but also well-prepared to contribute to advancements in technology. The course is structured to instill not only theoretical knowledge but also practical skills and a mindset of innovation, ensuring that graduates are highly employable in diverse and evolving roles such as software development, data analysis, and system administration.

Sign of the BOS Chairman
Dr. Jyotshna Dongardive
Ad-hoc BOS (Computer Science)

Sign of the Offg. Associate Dean
Dr. Madhav R. Rajwade
Faculty of Science & Technology

Sign of Offg. Dean
Prof. Shivram S. Garje
Faculty of Science & Technology