Detailed Syllabus of Minor Courses

Programme	B. Sc. Computer Science				
Course Code	CSC1MN101				
Course Title	Exploring Computer Basics & Computational Thinking				
Type of Course	**Minor**				
Semester	I				
Academic Level	100-199				
Course Details	Credit	Lecture per week	Tutorial per week	Practical per week	Total Hours
	4	3	-	2	75
Pre-requisites	1. Foundation on Mathematics at Plus Two level 2. Foundation on Basic Science at Plus Two Level				
Course Summary					

Course Outcomes (CO):

CO	CO Statement	Cognitive Level*	Knowledge Category#	Evaluation Tools used
CO1	Understanding of computer hardware, software, and basic operation principles	U	C	Exams/ Assignments/ Quizzes/ Seminars/ Practical
CO2	Understand and identify computer hardware components	U, Ap	C	Exam/ Assignments/ Quizzes/ Seminars/ v
CO3	Understand how data is represented and manipulated within a computer system.	U	C	Exam/ Assignments/ Quizzes/ Seminars
CO4	Understand the basics of computer languages, operating systems, and their comparison	U	C	Exam/ Assignments/ Quizzes/ Seminars

CO5	Learn to design and implement algorithms to solve simple computational problems.	U	P	Exam/ Assignments/ Quizzes/ Seminars/ / Practical
CO5	Develop computational thinking skills essential for problem-solving in various domains	Ap	P	Exam/ Assignments/ Quizzes/ Seminars/ / Practical

* - Remember (R), Understand (U), Apply (Ap), Analyse (An), Evaluate (E), Create (C)

- Factual Knowledge(F) Conceptual Knowledge (C) Procedural Knowledge (P) Metacognitive Knowledge (M)

Detailed Syllabus:

Module	Unit	Content	Hrs	Marks
I	**History, Evolution of Computers, and Number System**		**8**	**15**
	1	Introduction to Computers, Characteristics of Computers	1	
	2	Generations of Computers	1	
	3	Classification of Computers: Super Computers, Main Frame Computers, Mini Computers, Micro Computers	1	
	4	Number Systems (Binary, Decimal, Octal, Hexadecimal) and Conversion	3	
	5	Computer Codes: BCD Code, Excess 3 Code, ASCII Code, Unicode, Gray Code	2	
II	**Basic Computer Organization and Concept of Hardware**		**14**	**20**
	6	● Basic Computer Organization: Input Unit, Storage Unit, Processing Unit, Control Unit, Output Unit	1	
	7	● Concept of hardware	1	
	8	● CPU: Arithmetic and Logic Unit, Control unit	1	
	9	● Memory: Primary Memory, Secondary Memory, Access Time, Storage Capacity-bit, byte, nibble, volatile memory	2	
	10	● Memory hierarchy: Register memory, Cache memory, RAM (Static, Dynamic), ROM(Masked ROM, PROM and EPROM), Secondary storage devices (Magnetic tape, Hard disk, SSD and CD drive)	5	

		● Inside CPU: SMPS, Motherboard, Processor, Storage Devices (HDD, SSD, RAM, ROM).	1	
	11	● Motherboard Components: Processor Slot, Cooling Fan, RAM, Expansion Slots (PCIe), Input/Output Ports, Chipset, BIOS/UEFI Chip, SATA/NVMe Slots, Network Interface, Ports- Ethernet, VGA port, HDMI port, USB port.	3	
III	**Input-Output Devices, Concept of Software**		**12**	**15**
	12	● Input Devices: keyboard, pointing devices (mouse, touchpad), Video digitizer, remote control, joystick, scanner, digital camera, microphone, sensor	2	
	13	● Output Devices: monitor, printer (laser, inkjet, dot-matrix), plotter, speaker, control devices (lights, buzzers, robotic arms, and motors)	2	
	14	● Types of Software: System Software vs. Application Software, Proprietary Vs Open Source	2	
	15	Operating Systems: Functions, types of OS (batch, multiprogramming, time-sharing, real-time, and distributed)	2	
	16	Programming Languages (Machine, assembly & High level),	2	
	17	language Translators (Assembler, Interpreter and Compiler)	2	
1V	**Problem-solving and logical Thinking**		**11**	**20**
	18	Introduction to Problem Solving: Understanding the importance of problem-solving in computer science, Identifying and defining problems in a computational context.	2	
	19	Algorithm and its characteristics	1	
	20	Algorithm Development: Steps involved in designing algorithms, Pseudocode is an intermediate step in algorithm development.	2	
	21	Flowchart Basics: Introduction to flowcharts as a visual representation of algorithms, Understanding flowchart symbols and their meanings	2	
	22	Drawing simple flowcharts	4	
V	**Hands-on Data Structures:** **Practical Applications, Case Study and Course Project**		**30**	

	1	**Hardware:** 1. Identify the given motherboard components. 2. Identify and describe various ports and connectors on the motherboard.	5	
	2	**Software:** 1. Check the hardware compatibility and Install an operating system on a given computer. 2. Install any device driver on a given computer system to communicate with peripheral devices like Printers, Scanner, etc..	5	
	3	**Design Algorithm and visualize it using RAPTOR software** Problem 1: Calculate the Sum of Two Numbers Problem 2: Find the Larger of Two Numbers Problem 3: Check if a Number is Even or Odd Problem 4: Calculate the Factorial of a Number Problem 5: Temperature Conversion Problem 6: Simple Interest Calculation Problem 7: Calculate the Sum of Digits in a Number Problem 8: Check if a Number is Positive, Negative, or Zero Problem 9: Determine if a Triangle is Equilateral, Isosceles, or Scalene Problem 10: Check if a Number is Prime or Composite	20	

Reference Books:

1. Brookshear, J. Glenn. Computer Science: An Overview. 13th ed., Pearson, 2014.

2. Norton, Peter. Introduction to Computers. 7th ed., McGraw-Hill, 2016.

3. Patterson, David A. and John L. Hennessy. Computer Organization and Design: The Hardware/Software Interface. 5th ed., Morgan Kaufmann, 2013.

4. Sedgewick, Robert, and Kevin Wayne. Algorithms. 4th ed., Addison-Wesley Professional, 2011.

5. Knuth, Donald E. The Art of Computer Programming, Volumes 1-4A Boxed Set. Addison-Wesley Professional, 2011.

6. Grover, Aditya Bhargava. Grokking Algorithms: An Illustrated Guide for Programmers and Other Curious People. Manning Publications, 2016.

Mapping of COs with PSOs and POs :

	PSO1	PSO2	PSO3	PSO4	PSO5	PSO6	PO1	PO2	PO3	PO4	PO5	PO6
CO1	1	2	-	-	-	-	-					

CO 2	1	2	-	-	-	-	-					
CO 3	1	2	-	-	-	-	-					
CO 4	-	2	2	2	-	-	-					
CO 5	-	2	2	2	-	-	-					
CO 6	-	2	2	2	-	1	-					

Correlation Levels:

Level	Correlation
-	Nil
1	Slightly / Low
2	Moderate / Medium
3	Substantial / High

Assessment Rubrics:

- Quiz / Assignment/ Quiz/ Discussion / Seminar
- Midterm Exam
- Programming Assignments (20%)
- Final Exam (70%)

Mapping of COs to Assessment Rubrics:

	Internal Exam	Assignment	Project Evaluation	End Semester Examinations
CO 1		✓		✓
CO 2	✓	✓		✓
CO 3	✓	✓		✓
CO 4	✓	✓		✓
CO 5	✓	✓		✓

CO 6	✓	✓	✓	✓

Programme	B. Sc. Computer Science				
Course Code	CSC2MN101				
Course Title	Foundations of C Programming				
Type of Course	**Minor**				
Semester	II				
Academic Level	100-199				
Course Details	Credit	Lecture per week	Tutorial per week	Practical per week	Total Hours
	4	3	-	2	75
Pre-requisites	1. Basic Computer Literacy 2. Basic Problem-Solving Skills				
Course Summary	This course teaches the basics of programming using the C language. C is a powerful and widely used programming language known for its efficiency and flexibility. Through this course, students will learn how to write, understand, and debug C code to solve various problems and build simple applications.				

Course Outcomes (CO):

CO	**CO Statement**	**Cognitive Level***	**Knowledge Category#**	**Evaluation Tools used**
CO1	Demonstrate a solid understanding of fundamental programming concepts	An	P	Instructor-created lab exams / Quiz
CO2	Develop effective problem-solving skills by applying algorithmic thinking and logical reasoning.	An	P	Problem-solving assessments
CO3	Gain proficiency in writing, compiling, debugging, and executing C programs to implement algorithms, solve	Ap	P	Modeling Assignments

	problems, and create applications.			
CO4	Learn techniques to write efficient and optimized C code, including memory management, algorithm design, and performance tuning, to produce high-quality and scalable software solutions.	Ap	P	Modeling Assignments/ / Case studies
CO5	Understand and apply software development practices such as modular programming, code documentation, and debugging techniques to write maintainable and robust C programs.	Ap	P	Modeling Assignments/ / Case studies
CO6	Develop critical thinking skills by analyzing and evaluating C code, identifying errors and inefficiencies, and proposing solutions to improve code quality and performance.	Ap	P	Hands-on exercises

* - Remember (R), Understand (U), Apply (Ap), Analyse (An), Evaluate (E), Create (C)

- Factual Knowledge(F) Conceptual Knowledge (C) Procedural Knowledge (P) Metacognitive Knowledge (M)

Detailed Syllabus:

Module	Unit	Content	Hrs	Marks
I	**Problem-solving and logical Thinking**		**10**	**15**
	1	Overview of computational thinking concepts. Definition of algorithm and its characteristics.Understanding the importance of algorithms in problem-solving	2	
	2	Algorithm Development: Steps involved in designing algorithms	2	
	3	Pseudocode as an intermediate step in algorithm development.	1	

	4	Understanding flowchart symbols and their meanings.Learning to represent algorithms using flowcharts.	2	
	5	Raptor as a precursor to text-based programming languages	2	
	6	Drawing simple flowcharts	1	
II	**Introduction to C**		**10**	**20**
	7	Structure of C program	2	
	8	C Character Set, Keywords, Identifiers	1	
	9	Data Types, Variables, Declarations, Symbolic Constants	2	
	10	Operators:Arithmetic, Logical, Relational & Equality, and Unary, Operator Precedence and Associativity	2	
	11	Library Functions, Comments	1	
	12	I/O functions- Formatted scanf() & printf().	2	
III	**Control Statements, Arrays & Strings**		**14**	**20**
	13	Selection Statements:if, if-else, switch	3	
	14	iteration: while, do while, for	4	
	15	Arrays: One dimensional and Two Dimensional(introduction only)	3	
	16	Strings: Basic string handling functions	2	
	17	Structure:Definition, Processing-period Operator, Union(Concepts only)	2	
1V	**User-defined Functions**		**11**	**15**
	18	Definition of function, Advantages, Understanding function prototypes, and declarations	3	
	19	Introduction to function definitions and function calls	3	
	20	Exploring function parameters: actual and formal parameters	2	
	21	Recursion	2	
	22	Pointers-declarations(Basic concept only)	1	
V	**Hands-on C:** **Practical Applications, Case Study and Course Project**		**30**	

	1	Write a C program using Variables and Data Types Write a C program using Arithmetic Operations Write a C program using Loops Write a C program using Arrays Write a C program using Functions Write a C program using Strings	20	
	2	Case study: 1. Library Management System: Develop a program to manage a library's collection of books. Implement functions for adding, removing, and searching for books. 2. Ticket Booking System: Design a program to manage ticket bookings for a cinema or theater.	5	
	3	Capstone/Course Project: Design a real-time project in C	5	

Reference:

1. Balagurusamy, E. Programming in ANSI C. Tata McGraw-Hill Education, 2019.
2. King, K. N. C Programming: A Modern Approach. 2nd ed., W. W. Norton & Company, 2008.
3. Kernighan, Brian W., and Dennis M. Ritchie. The C Programming Language. 2nd ed., Prentice Hall, 1988.
4. Prata, Stephen. C Primer Plus. 6th ed., Addison-Wesley, 2013.
5. Perry, Greg. Absolute Beginner's Guide to C. 3rd ed., Que Publishing, 2014.
6. Oualline, Steve. Practical C Programming. 3rd ed., O'Reilly Media, 1997.
7. Hanly, Jeri R., and Elliot B. Koffman. Problem-Solving and Program Design in C. 8th ed., Pearson, 2016.
8. Gottfried, Byron S. Programming with C. 2nd ed., McGraw-Hill, 1996.
9. Holmes, Dan. C in a Nutshell. 2nd ed., O'Reilly Media, 2015.

Mapping of COs with PSOs and POs :

	PSO1	PSO2	PSO3	PSO4	PSO5	PSO6	PO1	PO2	PO3	PO4	PO5	PO6
CO1	3	1	-	-	-	1						
CO2	1	-	2	-	-	-						
CO3	-	-	2	-	-	-						
CO4	-	1	3	3	-	3						
CO5	-	2	3	3	-	3						
CO6	-	-	-	-	-	3						

Correlation Levels:

Level	**Correlation**
-	Nil
1	Slightly / Low
2	Moderate / Medium
3	Substantial / High

Assessment Rubrics:

- Quiz / Assignment/ Quiz/ Discussion / Seminar
- Midterm Exam
- Programming Assignments (20%)
- Final Exam (70%)

Mapping of COs to Assessment Rubrics :

	Internal Exam	Assignment	Project Evaluation	End Semester Examinations
CO 1		✓		✓
CO 2	✓	✓		✓
CO 3		✓		✓
CO 4	✓			✓
CO 5	✓		✓	✓
CO 6	✓		✓	✓

Programme	B. Sc. Computer Science				
Course Code	CSC3MN201				
Course Title	Python Programming				
Type of Course	**Minor**				
Semester	III				
Academic Level	200-299				
Course Details	Credit	Lecture per week	Tutorial per week	Practical per week	Total Hours
	4	3	-	2	75
Pre-requisites	Have an understanding of algorithms and flowcharts				
Course Summary	This course explores the versatility of Python language in programming and teaches the application of various data structures using Python.				

Course Outcomes (CO):

CO	CO Statement	Cognitive Level*	Knowledge	Evaluation Tools used
CO1	Understand the basic concepts of Python programming	U	C	Instructor- created exams / Quiz
CO2	Apply problem-solving skills using different control structures and loops	Ap	P	Coding Assignments/ Code reading and review
CO3	Design simple Python programs to solve basic computational problems and acquire knowledge of Python's error-handling mechanisms to effectively debug programs	Ap	P	Coding Assignments/ Exams
CO4	Analyze the various data structures and operations on it using Python	An	P	Instructor-created exams / Case studies
CO5	Apply modular programming using functions	U	C	Instructor- created exams / Quiz

CO6	Identify the necessary Python packages in the domain and create simple programs with it	U, Ap	C, P	Coding

* - Remember (R), Understand (U), Apply (Ap), Analyse (An), Evaluate (E), Create (C)

- Factual Knowledge(F) Conceptual Knowledge (C) Procedural Knowledge (P) Metacognitive Knowledge (M)

Detailed Syllabus:

Module	Unit	Content	Hrs	Marks
I	**Introduction to Python**		**12**	15
	1	Features of Python, Different methods to run Python,Python IDE	2	
	2	Comments, Indentation, Identifiers, Keywords, Variables	2	
	3	Standard Data Types	2	
	4	Input Output Functions, Import Functions, range function	1	
	5	Operators and Operands, Precedence of Operators, Associativity	2	
	6	Type Conversion, Multiple Assignment	1	
	7	Expressions and Statements, Evaluation of Expressions	1	
	8	Boolean Expressions	1	
II	**Control Structures**		**12**	20
	9	Decision Making- if statement, if…else statement, if…elif…else statement, Nested if statement	5	
	10	Loops - for loop, for loop with else, while loop, while loop with else, Nested Loops	5	
	11	Using indentation in Python to define code blocks	1	
	12	Control Statements- break, continue, pass	1	

III	**Data Structures in Python**		**12**	15
	13	Working with strings and string manipulation	3	
	14	List - creating list, accessing, updating and deleting elements from a list	2	
	15	Basic list operations	1	
	16	Tuple- creating and accessing tuples in python	2	
	17	Basic tuple operations	1	
	18	Dictionary, built in methods to create, access, and modify key-value pairs	2	
	19	Set and basic operations on a set	1	
	Functions		**9**	
IV	20	Built-in functions - mathematical functions, date time functions, random numbers	1	20
	21	Writing user defined functions - function definition, function call, flow of execution, parameters and arguments, return statement	6	
	22	Recursion. Introduction to basic Python libraries (e.g., math, random)	2	
	Hands-on Data Structures: **Practical Applications, Case Study and Course Project**		**30**	
Design programs from the concepts listed below. Select the topics and programs suited for your domain				
V	1	Programs to: • Run instructions in Interactive interpreter and as Python Script • Perform calculations involving integers and floating point numbers using Python arithmetic operators **Data Structures in Python** • String - Create a string , Indexing / Looping / Slicing		

		• Lists - Create a list , Indexing /Looping / Slicing , Adding items / Modifying items / Removing items • Tuples - Create a tuple , Indexing / Looping / Slicing / Adding items to a tuple • Dictionary - Create a dictionary and access values with key / Adding a key- value pair / Adding to an empty dictionary /Modifying values in a dictionary / Removing key-value pair **Function** • Call functions residing in the math module • Define a function for later use • Pass one or more values into a function • Return one or more results from a function		
		Case study: • Create a Todo List Manager where Users should be able to add, remove, and view tasks • Create Student Grade Tracker: Allow users to add students, add grades for subjects, and calculate average grades.		

Reference Books:

1. Jose, Jeeva. Taming Python By Programming. Khanna Book Publishing, 2017. Print.
2. Downey, Allen. Think Python. Green Tea Press, 2nd ed. 2009

Mapping of COs with PSOs and POs :

	PSO1	PSO 2	PSO 3	PSO4	PSO5	PSO6	PO 1	PO2	PO3	PO4	PO5	PO6
CO 1	-	1	2	3	1	1						
CO 2	-	1	2	3	1	1						
CO	-	2	2	3	1	1						

3												
CO 4	1	1	-	-	1	-						
CO 5	1	1	2	2	1	-						
CO 6	-	1	2	2	2	1						

Correlation levels

Level	**Correlation**
-	Nil
1	Slightly / Low
2	Moderate / Medium
3	Substantial / High

Assessment Rubrics:

- Quiz / Assignment/ Quiz/ Discussion / Seminar
- Midterm Exam
- Programming Assignments (20%)
- Final Exam (70%)

Mapping of COs to Assessment Rubrics :

	Internal Exam	Assignment	Project Evaluation	End Semester Examinations
CO 1	✓			✓
CO 2	✓	✓	✓	✓
CO 3	✓		✓	✓
CO 4	✓	✓	✓	✓
CO 5	✓			✓
CO 6	✓			✓