AC/27.06.2023/RS1

College of Arts, Science & Commerce (Autonomous)

SIES

RISE WITH EDUCATION

NAAC REACCREDITED - 'A' GRADE

SIES College of Arts, Science and Commerce (Autonomous)

Affiliated to University of Mumbai

Syllabus under NEP effective from June 2023

Programme: B.Sc.

Subject: Information Technology

Core Course

Class: FYBSc(IT)

Semester : I and II

Choice Based Credit System (CBCS)
with effect from the academic year 2023-24

Semester I
Core Course

This Core course is offered to students of BSc(IT) in Semester I, who have chosen Information Technology as Major & Minor subject

Name of Programme: Bachelor of Science **Subject:** Information Technology						
Class	**Semester**	**Course Code**	**Course Name**	**No. of Lectures/ Practicals per week**	**Credits**	**Marks**
FYBSc(IT)	I	SIUITMJ111	Imperative Programming	3L	3	75
FYBSc(IT)	I	SIUITMJP111	Imperative Programming Practical	1P per batch	1	25
P (Practical) = 2 Hours per week						

Course Name: Imperative Programming **Credits: 3 Type: Theory**		
Expected Course Outcomes On completion of this course, students will be able to 1. Remember the data types, structure of if statement, and the loops in C language 2. Write programs using if-else structure, loops, switch statement and user defined functions. 3. Create and use one and two dimensional arrays in programs and use basic pointers in C.		
Pre-requisites:	Basics of Mathematics	
Unit I	Basics of C	15 Lectures
	● What is C? ● Data Types ● The Decision Control Structure, The if Statement, The if-else Statement, Use of Logical Operators ● The Case Control Structure	
Unit II	Loops	15 Lectures
	● The Loop Control Structure, The while Loop, The for Loop, The Odd Loop, The do-while Loop ● Storage Classes , Automatic Storage Class, Register Storage Class, Static Storage Class, External Storage Class	

Unit III	Functions	15 Lectures
	● Arrays, 1D and 2D Arrays ● Pointers ● Functions, Passing Values between Functions, Call by Value and Call by Reference, ● Puppetting on Strings, Standard Library String Functions ● Structures, Declaring a Structure, Accessing Structure Elements, How Structure Elements are Stored	

Course Name: Imperative Programming Practical **Credits: 1 Type: Practical**	
Expected Course Outcomes On completion of this course, students will be able to 1. Apply the use of if structure, loops and functions and execute them by writing programs. **2.** Test the use of pointers, arrays and strings by coding in C language.	
Practical No.	Title
01	a. Write a program to swap 2 numbers using 2 and 3 variables. b. Write a program to find the area of rectangle, square and circle. c. Write a program to find the volume of a cube, sphere, and cylinder.
02	a. Write a program to enter a number from the user and display the month name. If number >=13 then display invalid input using switch case. b. Write a program to check whether the number is even or odd. c. Write a program to check whether the number is positive, negative or zero. d. Write a program to find the factorial of a number. e. Write a program to check whether the entered number is prime or not. f. Write a program to find the largest of three numbers.
03	a. Write a program to find the sum of squares of digits of a number. b. Write a program to find the sum of numbers from 1 to 100. c. Write a programs to print the Fibonacci series. d. Write a program to find the reverse of a number and check if it is a palindrome or not. e. Write a program to solve a quadratic equation. f. Write a program to check whether the entered number is Armstrong or not. g. Write a program to count the digits in a number.
04	a. Write a program to check whether a given number is prime number or not. Use user defined functions. b. Write a program to find the sum of the digits of a number. Use user defined functions. c. Write a program to swap 2 numbers. Use call by value and call by reference for the same
05	a. Write a program to find the factorial of a number using recursive function. b. Write a program to find the sum of natural number using recursive function.
06	a. Write a program to find the largest value that is stored in the array. b. Write a program using pointers to compute the sum of all elements stored in an array. c. Write a program to arrange the ‘n’ numbers stored in the array in ascending and descending order. d. Write a program that performs addition and subtraction of matrices.

	e. Write a program that performs multiplication of matrices.
07	a. Write a program to swap 2 numbers using call by reference. b. Write a program to find area and perimeter of a circle using call by reference
08	Write a program to find the length of a string, concatenate 2 strings, copy a string from one location to the other using built-in and user defined functions
09	Write a program to accept 3 details of books and print the same using structures.
10	Programs on different patterns. (any 2 patterns)

References
1. Let Us C: Authentic guide to C programming language, by Yashvant Kanetkar, BPB Publications, 19th Edition, 2022

Scheme of Evaluation:

I) Continuous Internal Evaluation (25 Marks)	
Class Test	10 Marks
Assignment/ Project and Viva/ Presentation	15 Marks
II) Theory Examination (50 Marks)	
Semester End Examination based on entire syllabus	50 Marks
III) Practical Examination (25 Marks)	
Certified Journal	5 Marks
Viva Voce	5 Marks
Practical exam	15 Marks

Semester I
Core Course

This Core course is offered to students of BSc(IT) in Semester I, who have chosen Information Technology as Major & Minor subject

Name of Programme: Bachelor of Science			**Subject:** Information Technology			
Class	**Semester**	**Course Code**	**Course Name**	**No. of Lectures/ Practicals per week**	**Credit s**	**Marks**
FYBSc(IT)	I	SIUITMN111	Digital Electronics	3L	3	75
FYBSc(IT)	I	SIUITMNP111	Digital Electronics Practical	1P per batch	1	25
P (Practical) = 2 Hours per week						

Course Name: Digital Electronics **Credits: 3 Type: Theory**		
Expected Course Outcomes On completion of this course, students will be able to 1. Understand the structure of various number systems, binary arithmetic and itsapplications in digital design. 2. Apply the Boolean algebra using logic gates and Karnaugh Maps. 3. Construct and design Combinational and Sequential Logic circuits.		
Pre-requisites:	None	
Unit I	Number System and Binary Arithmetic	15 Lectures
	Analog System, digital system, numbering system, binary number system, octal number system, hexadecimal number system, conversion from onenumber system to another, floating point numbers, weighted codes binary coded decimal, non-weighted codes Excess – 3 code, Gray code. Binary addition, Binary subtraction, Negative number representation, Subtraction using 1’s complement and 2’s complement.	
Unit II	Boolean Algebra, Logic Gates and Karnaugh Maps	15 Lectures

	Introduction, Logic (AND OR NOT), Boolean theorems, Boolean Laws, De Morgan's Theorem, Reduction of Logic expression using Boolean Algebra, Deriving Boolean expression from given circuit, exclusive OR and ExclusiveNOR gates, Universal Logic gates, Implementation of other gates using universal gates. Introduction, minterms and sum of minterm form, maxterm and Product of maxterm form, Reduction technique using Karnaugh maps – 2/3/4 variables K-maps, Grouping of variables in K-maps, K-maps for product of sum form,minimize Boolean expression using K-map and obtain K-map from Booleanexpression.	
Unit III	Combinational Circuits and Sequential Circuits.	15 Lectures
	Introduction, Multi-input, multi-output Combinational circuits, Code Converters design and implementations. Introduction, Multiplexer, Demultiplexer, Decoder, Encoders. Introduction, Terminologies used, S-R flip-flop, D flip-fop, JK flip-flop, Race-around condition, Master – slave JK flip-flop, T flip-flop.	

Course Name: Digital Electronics Practical **Credits: 1 Type : Practical**	
Expected Course Outcomes On completion of this course, students will be able to 1. Design Sequential and Combinational circuits. 2. Simplify the Logistical expressions.	
Practical No.	Title
1.	**Study of Logic gates and their ICs and universal gates:**
	a. Study of AND, OR, NOT, XOR, XNOR, NAND and NOR gates b. IC 7400, 7402, 7404, 7408, 7432, 7486, 74266 c. Implement AND, OR, NOT, XOR, XNOR using NAND gates. d. Implement AND, OR, NOT, XOR, XNOR using NOR gates.
2.	**Implement the given Boolean expressions using minimum number of gates.**
	a. Verifying De Morgan's laws. b. Implement other given expressions using minimum number of gates. c. Implement other given expressions using minimum number of ICs.
3.	**Implement combinational circuits.**
	Design and implement combinational circuit based on the problem given and minimizing using K-maps.
4.	**Implement code converters.**
	a. Design and implement Binary – to – Gray code converter. b. Design and implement Gray – to – Binary code converter. c. Design and implement Binary – to – BCD code converter. d. Design and implement Binary – to – XS-3 code converter.
5.	**Implement Encode and Decoder and Multiplexer and Demultiplexers.**
	a. Design and implement 8:3 encoder. b. Design and implement 3:8 decoder. c. Design and implement 4:1 multiplexer. Study of IC 74153, 74157 d. Design and implement 1:4 demultiplexer. Study of IC 74139 e. Implement the given expression using IC 74151 8:1 multiplexer. f. Implement the given expression using IC 74138 3:8 decoder.
6.	**Study of flip-flops and counters.**
	a. Study of IC 7473. b. Study of IC 7474. c. Study of IC 7476. d. Conversion of Flip-flops. e. Design of 3-bit synchronous counter using 7473 and required gates. f. Design of 3-bit ripple counter using IC 7473.
7.	**Design of shift registers and shift register counters.**
	a. Design serial – in serial – out, serial – in parallel – out, parallel – in serial – out, parallel – in parallel – out and bidirectional shift registers using IC 7474. b. Implementation of digits using seven segment displays.

References

1. Digital Electronics and Logic Design, N. G. Palan , 1st Edition,
2. Make Electronics, Charles Platt, 1se Edition
3. Modern Digital Electronics, R. P. Jain , 4th Edition
4. Digital Principles and Applications , Malvino and Leach, 8th Edition.

Scheme of Evaluation:

I) Continuous Internal Evaluation (25 Marks)	
Class Test	10 Marks
Assignment/ Project and Viva/ Presentation	15 Marks
II) Theory Examination (50 Marks)	
Semester End Examination based on entire syllabus	50 Marks
III) Practical Examination (25 Marks)	
Certified Journal	5 marks
Viva Voce	5 marks
Practical exam	15 marks

Semester II
Core Course

This Core course is offered to students of BSc(IT) in Semester II, who have chosen Information Technology as Major & Minor subject

Name of Programme: Bachelor of Science **Subject:** Information Technology						
Class	**Semester**	**Course Code**	**Course Name**	**No. of lectures/ practical per week**	**Credits**	**Marks**
FYBSc(IT)	II	SIUITMJ121	Object Oriented Programming with C++	3L	3	75
FYBSc(IT)	II	SIUITMJP121	Object Oriented Programming with C++ Practical	1P per batch	1	25
P(Practical)=2 Hours per week						

Course Name: Object Oriented Programming with C++ **Credits: 3 Type: Theory**		
Expected Course Outcomes On completion of this course, students will be able to 1. Explain the difference between top down and bottom up approach in programming andoutline the essential features and elements of C++ programming. 2. Apply the concepts of function,operator overloading and inheritance. Incorporate exception handling in object oriented programs and use template classes and standardlibrary in C++.		
Pre-requisites:	Basics of C	
Unit I	● Object oriented and Procedure oriented Language, Object oriented theme, development, benefits and applications of OOPs ● Concepts of OOPS: Objects, Classes, Data Abstraction and Data Encapsulation, Inheritance, Polymorphism ● Classes and Objects: Simple Class structure, Defining member functions inside and outside class, passing object as an argument, Returning object from functions, friend function.	15 Lectures

	● Constructors and Destructors: Introduction, Default Constructor, Parameterized Constructor and examples, copy constructor, Destructors.	
Unit II	● Polymorphism: Concept of Function overloading, overloaded operators, overloading unary and binary operators, overloading comparison operator, overloading arithmetic assignment operator. ● Inheritance: Introduction, understanding inheritance, Advantages provided by inheritance, choosing the access specifier, Derived class declaration, derived class constructors, class hierarchies, multiple inheritance, multilevel inheritance, hybrid inheritance. ● Virtual Functions & Abstract Class: Introduction and need, Pure Virtual Functions, Static Functions, this Pointer, abstract classes, virtual destructors	15 Lectures
Unit III	● String Handling: Introduction, creating string objects, string characteristics, manipulating string. ● Exception Handling: Introduction, Exception Handling Mechanism, Concept of throw & catch with example ● Templates: Introduction, Function Template and examples, Class Template and examples. ● Working with Files: Introduction, File Operations, Various File Modes, File Pointer and their Manipulation.	15 Lectures

Course Name: Object Oriented Programming with C++ Practical
Credits: 1 Type: Practical

Expected Course Outcomes

On completion of this course, students will be able to

1. Apply use of class and object, constructor and destructor and execute them with programs.
2. Write programs with polymorphism, inheritance, and abstract class.
3. Write programs with handling of strings, exception and files.

Practical No.	Title
01	**Classes and methods**
	a. Design an employee class for reading and displaying the employee information, the getInfo() and displayInfo() methods will be used repectively. Where getInfo()will be private method b. Design the class student containing getData() and displayData() as two of its methods which will be used for reading and displaying the student informationrespectively.WheregetData() will be private method. c. Design the class Demo which will contain the following methods: readNo(), factorial() for calculating the factorial of a number, reverseNo() will reverse thegiven number, isPalindrome() will check the given number is palindrome, isArmstrong() which will calculate the given number is armStrong or not.WherereadNo() will be private method. d. Write a program to demonstrate function definition outside class and accessingclass members in function definition.
2.	**Using friend functions**
	a. Write a friend function for adding the two complex numbers, using a single class b. Write a friend function for adding the two different distances and display its sum,using two classes. c. Write a friend function for adding the two matrix from two different classes anddisplay its sum.
3.	**Constructors and method overloading.**
	a. Design a class Complex for adding the two complex numbers and also show theuse of constructor b. Design a class Geometry containing the methods area() and volume() and alsooverload the area() function. c. Design a class StaticDemo to show the implementation of static variable andstatic function.
4.	**Operator Overloading**
	a. Overload the operator unary(-) for demonstrating operator overloading b. Overload the operator + for adding the timings of two clocks, And

	also passobjects as an argument. c. Overload the + for concatenating the two strings. For e.g “Py” + “thon” = Python
5.	**Inheritance**
	a. Design a class for single level inheritance using public and private type Derivation. b. Design a class for multiple inheritance. c. Implement the hierarchical inheritance
6.	**Virtual functions and abstract classes**
	a. Implement the concept of method overriding. b. Show the use of virtual function c. Show the implementation of abstract class.
7.	**String handling**
	a. String operations for string length , string concatenation b. String operations for string reverse, string comparison, c. Console formatting functions.
8.	**Exception handling**
	a. Show the implementation of exception handling b. Show the implementation for exception handling for strings c. Show the implementation of exception handling for using the pointers.
9.	**File handling**
	a. Design a class FileDemo open a file in read mode and display the total number ofwords and lines in the file. b. Design a class to handle multiple files and file operations c. Design a editor for appending and editing the files
10.	**Templates**
	a. Show the implementation for template function b. Show the implementation of template class library for swap function. c. Design the template class library for sorting ascending to descending and viceversa

References

1. Object Oriented Programming with C++ by E Balagurusamy, TMH Publications, 7th Edition, 2017.
2. C++ for beginners by B. M. Hirwani, SPD Publication, 2013.
3. Mastering C++ by, K R Venugopal, RajkumarBuyya, T Ravishankar, TMH Publication, 2nd Edition ,2011.

Scheme of Evaluation:

I) Continuous Internal Evaluation (25 Marks)	
Class Test	10 Marks
Assignment/ Project and Viva/ Presentation	15 Marks
II) Theory Examination (50 Marks)	
Semester End Examination based on entire syllabus	50 Marks
III) Practical Examination (25 Marks)	
Certified Journal	5 marks
Viva Voce	5 marks
Practical exam	15 marks

Semester II
Core Course

This Core course is offered to students of BSc(IT) in Semester II, who have chosen Information Technology as Major & Minor subject

Name of Programme: Bachelor of Science			**Subject:** Information Technology			
Class	**Semester**	**Course Code**	**Course Name**	**No. of lectures/ practical per week**	**Credits**	**Marks**
FYBSc(IT)	II	SIUITMN121	Discrete Mathematics	3L	3	75
FYBSc(IT)	II	SIUITMNP121	Discrete Mathematics Practical	1P per batch	1	25
P (Practical) = 2 Hours per week						

Course Name: Discrete Mathematics **Credits: 3 Type: Theory**		
Expected Course Outcomes On completion of this course, students will be able to 1. Describe concepts of set theory, conditional statements, and identify valid and invalid arguments 2. Explain the significance of quantified statements and describe sequences, mathematicalinduction and recursion in Mathematics. 3. Classify relations, graphs and trees, implement functions on general sets and solve problems related to counting and probability.		
Pre-requisites:	Basics of Mathematics	
Unit I	**Set Theory and Functions**	15 Lectures
	● **Set Theory**: Definitions, Properties of Sets, Venn Diagram, Problemson sets, Principle of Inclusion and Exclusion (only statement), problems. ● **Functions**: Functions Defined on General Sets, One-to-One and Onto,Inverse Functions, Composition of Functions, Floor and Ceiling.	
Unit II	**Sequences, Mathematical Induction, and Recursion**,	15

	Relations	Lectures
	● **Sequences, Mathematical Induction, and Recursion**: Sequences, Mathematical Induction, defining sequences recursively, solving recurrence relations by iteration, Second order linear homogenous recurrence relations with constant coefficients. ● **Relations**: Relations on Sets, Reflexivity, Symmetry, and Transitivity,Equivalence Relations, Partial Order Relations, Digraph.	
Unit III	**Graphs and Trees, Probability and Pigeon hole Principle**	15 Lectures
	● **Probability and Pigeon hole Principle:** Introduction, Addition rule, Product rule, Conditional probability, permutations and combinations, pigeon hole principle. ● **Graphs and Trees**: Definitions and Basic Properties, Trails, Paths,and Circuits, Matrix Representations of Graphs, Isomorphism's of Graphs, Trees, Rooted Trees, Isomorphism's of Graphs, Spanning trees and shortest paths.	

Course Name: Discrete Mathematics Practical
Credits: 1 Type: Practical

Expected Course Outcomes

On completion of this course, students will be able to

1. Apply the use of set theory and functions by writing programs in Scilab.
2. Relate the use of Graphs, Trees and Probability by writing codes in Scilab.

Practical No.	Title
01	a. Write a program to find **e^x**. $e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \dots$ b. Write a program to solve a given quadratic equation. c. The natural logarithm(log$_{e)}$ can be approximated by the following series: $\frac{x-1}{x} + \frac{1}{2}\left(\frac{x-1}{x}\right)^2 + \frac{1}{2}\left(\frac{x-1}{x}\right)^3 + \frac{1}{2}\left(\frac{x-1}{x}\right)^4 + \dots.$ If x is input through the keyboard, write a program to calculate the sum of first seven terms of this series. d. Write a program to check equilateral, scalene or isosceles triangles. If side1, side2 and side3 are the 3 sides of a triangle taken as input then: i. Check if(side1 == side2 && side2 == side3), then the triangle is equilateral. ii. If it is not an equilateral triangle then it may be isosceles. Check if(side1 == side2 || side1 == side3 || side2 == side3), then the triangle is isosceles. iii. If it is neither equilateral nor isosceles then it is a scalene triangle.
02	a. Calculation of 20 terms of a sequence defined by recurrence by: $\begin{cases} u_1 = 4 \\ u_{n+1} = u_n + 2n + 3 \end{cases}$ b. I planted a Christmas tree in 2005 measuring 1.20 m. It grows by 30 cm per year. I decided to cut it when it exceeds 7 m. In what year will I cut the tree? c. Alice throws three dice. • If she gets three 6's she wins $20, • If she gets three identical numbers different from 6 she wins $10, • If she gets two identical numbers she wins $5, • Otherwise she wins nothing. Simulate a trial and calculate Alice's winnings d. Write a program in C to display the sum of the series [9 + 99 + 999 + 9999 ...]

03	a. Creating a Matrix, Accessing Elements In a Matrix, Accessing The Last Element Of a Matrix, Finding Last Element Of First Column of a Matrix, Finding Last Element Of Each Column of a Matrix, Adding A New Row To a Matrix, Finding Size Of a Matrix, Finding Square And Cube Of a Matrix b. Addition And Subtraction of Matrices, Matrix Multiplication, Determinant Of a Matrix, Inverse Of a Matrix, Functions Related To Matrix Creation c. Solving Linear Equation Using Matrices
04	**Set Theory -** Inclusion Exclusion principle, Power Sets, Mathematical Induction
05	**Functions and Algorithms -** Recursively defined functions, Cardinality, Polynomial evaluation, Greatest Common Divisor
06	**Counting -** Sum rule principle, Product rule principle, Factorial, Binomial coefficients, Permutations, Permutations with repetitions, Combinations, Combinations with repetitions, Ordered partitions, Unordered partitions
07	**Probability Theory -** Sample space and events, Finite probability spaces, Equiprobable spaces, Addition Principle, Conditional Probability, Multiplication theorem for conditional probability, Independent events, Repeated trials with two outcomes
08	**Graph Theory -** Paths and connectivity, Minimum spanning tree, Isomorphism
09	**Directed Graphs -** Adjacency matrix, Path matrix

References

1. Discrete Mathematics with Applications, Sussana S. Epp, Published by Cengage Learning, 4th Edition, 2010
2. Discrete Mathematics, Schaum’s Outlines Series, Seymour Lipschutz, Marc Lipson, Tata MCGraw Hill
3. Discrete Mathematics andits Applications, Kenneth H. Rosen, Tata MCGraw Hill

Scheme of Evaluation:

I) Continuous Internal Evaluation (25 Marks)	
Class Test	10 Marks
Assignment/ Project and Viva/ Presentation	15 Marks
II) Theory Examination (50 Marks)	
Semester End Examination based on entire syllabus	50 Marks
III) Practical Examination (25 Marks)	
Certified Journal	5 Marks
Viva voce	5 Marks
Practical exam	15 marks