DEPARTMENT OF COMPUTER SCIENCE
B.Sc. (H) Computer Science

CATEGORY-I

DISCIPLINE SPECIFIC CORE COURSE – 1: PROGRAMMING USING PYTHON

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
Programming using Python	4	3	0	1	Class XII pass	Nil

Learning Objectives

This course is designed as the first course that:

- Introduces programming concepts using Python to Computer Science students.
- Focuses on the development of Python programming to solve problems of different domains.
- Introduces the concept of object- oriented programming.

Learning Outcomes:

On successful completion of the course, students will be able to:

- Understand the basics of programming language
- Develop, document, and debug modular Python programs.
- Apply suitable programming constructs and built-in data structures to solve a problem.
- Use and apply various data objects in Python.
- Use classes and objects in application programs and handle files.

SYLLABUS OF DSC-1

Theory

Unit – 1 **(6 hours)**

Introduction to Programming

Problem solving strategies; Structure of a Python program; Syntax and semantics; Executing simple programs in Python.

Unit – 2 **(12 hours)**

Creating Python Programs

Identifiers and keywords; Literals, numbers, and strings; Operators; Expressions; Input/output statements; Defining functions; Control structures (conditional statements, loop control statements, break, continue and pass, exit function), default arguments.

Unit – 3 **(15 hours)**

Built-in Data Structures

Mutable and immutable objects; Strings, built-in functions for string, string traversal, string operators and operations; Lists creation, traversal, slicing and splitting operations, passing list to a function; Tuples, sets, dictionaries and their operations.

Unit – 4 **(6 hours)**

Object Oriented Programming

Introduction to classes, objects and methods; Standard libraries.

Unit – 5 **(6 hours)**

File and Exception Handling

File handling through libraries; Errors and exception handling.

Practical **(30 hours)**

List of Practicals:

1. WAP to find the roots of a quadratic equation
2. WAP to accept a number 'n' and
    j. Check if 'n' is prime
    k. Generate all prime numbers till 'n'
    l. Generate first 'n' prime numbers This program may be done using functions
3. WAP to create a pyramid of the character '*' and a reverse pyramid

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

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

8. WAP that accepts a character and performs the following:
    a. print whether the character is a letter or numeric digit or a special character
    b. if the character is a letter, print whether the letter is uppercase or lowercase
    c. if the character is a numeric digit, prints its name in text (e.g., if input is 9, output is NINE)

9. WAP to perform the following operations on a string
    a. Find the frequency of a character in a string.
    b. Replace a character by another character in a string.
    c. Remove the first occurrence of a character from a string.
    d. Remove all occurrences of a character from a string.
10. WAP to swap the first n characters of two strings.
11. Write a function that accepts two strings and returns the indices of all the occurrences of the second string in the first string as a list. If the second string is not present in the first string then it should return -1.
12. WAP to create a list of the cubes of only the even integers appearing in the input list (may have elements of other types also) using the following:
    a. 'for' loop
    b. list comprehension
13. WAP to read a file and
    m. Print the total number of characters, words and lines in the file.
    n. Calculate the frequency of each character in the file. Use a variable of dictionary type to maintain the count.
    o. Print the words in reverse order.
    p. Copy even lines of the file to a file named 'File1' and odd lines to another file named 'File2'.
14. WAP to define a class Point with coordinates x and y as attributes. Create relevant methods and print the objects. Also define a method distance to calculate the distance between any two point objects.
15. Write a function that prints a dictionary where the keys are numbers between 1 and 5 and the values are cubes of the keys.
16. Consider a tuple t1=(1, 2, 5, 7, 9, 2, 4, 6, 8, 10). WAP to perform following operations:
    a. Print half the values of the tuple in one line and the other half in the next line.
    b. Print another tuple whose values are even numbers in the given tuple.
    c. Concatenate a tuple t2=(11,13,15) with t1.
    d. Return maximum and minimum value from this tuple
17. WAP to accept a name from a user. Raise and handle appropriate exception(s) if the text entered by the user contains digits and/or special characters.

Essential Readings

- Taneja, S., Kumar, N. Python Programming- A modular Approach, 1st edition, Pearson Education India, 2018.

- Balaguruswamy E. Introduction to Computing and Problem Solving using Python, 2nd edition, McGraw Hill Education, 2018.

Suggestive Readings

- Brown, Martin C. Python: The Complete Reference, 2nd edition, McGraw Hill Education, 2018.
- Guttag, J.V. Introduction to computation and programming using Python, 2nd edition, MIT Press, 2016.

Note: Examination scheme and mode shall be as prescribed by the Examination Branch, University of Delhi, from time to time.

DISCIPLINE SPECIFIC CORE COURSE – 2: COMPUTER SYSTEM ARCHITECTURE

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
Computer System Architecture	4	3	0	1	Class XII pass	NIL

Learning Objectives

The Learning Objectives of this course are as follows:

- Introduces the students to the fundamental concepts of digital computer organization, design and architecture.
- Develop a basic understanding of the building blocks of the computer system and highlights how these blocks are organized together to architect a digital computer system.

Learning Outcomes

On successful completion of the course, students will be able to:

- Design Combinational Circuits using basic building blocks. Simplify these circuits using Boolean algebra and Karnaugh maps. Differentiate between combinational circuits and sequential circuits.
- Represent data in binary form, convert numeric data between different number systems and perform arithmetic operations in binary.
- Determine various stages of instruction cycle, pipelining and describe interrupts and their handling.
- Explain how CPU communicates with memory and I/O devices and distinguish between different types of processors.
- Simulate the design of a basic computer using a software tool.

SYLLABUS OF DSC - 2

Theory

Unit – 1 **(6 hours)**

Digital Logic Circuits

Logic Gates, Truth Tables, Boolean Algebra, Digital Circuits, Combinational Circuits, Introduction to Sequential Circuits, Circuit Simplification using Karnaugh Map, Don't Care Conditions, Flip-Flops, Characteristic Tables, Excitation Table.

Unit – 2 **(9 hours)**

Digital Components (Fundamental building blocks)

Designing of combinational circuits- Half Adder, Full Adder, Decoders, Encoders, Multiplexers, Registers and Memory (RAM, ROM and their types), Arithmetic Microoperations, Binary Adder, Binary Adder-Subtractor.

Unit – 3 **(6 hours)**

Data Representation and Basic Computer Arithmetic

Number System, r and (r-1)'s Complements, data representation and arithmetic operations.

Unit – 4 **(9 hours)**

Basic Computer Organization and Design

Bus organization, Microprogrammed vs Hardwired Control, Instruction Codes, Instruction Format, Instruction Cycle, Instruction pipelining, Memory Reference, Register Reference and Input Output Instructions, Program Interrupt and Interrupt Cycle..

Unit – 5 **(6 hours)**

Processors

General register organization, Stack Organization, Addressing Modes, Overview of Reduced Instruction Set Computer (RISC), Complex Instruction Set Computer (CISC), Multicore processor and Graphics Processing Unit (GPU).

Unit – 6 **(9 hours)**

Memory and Input-Output Organization

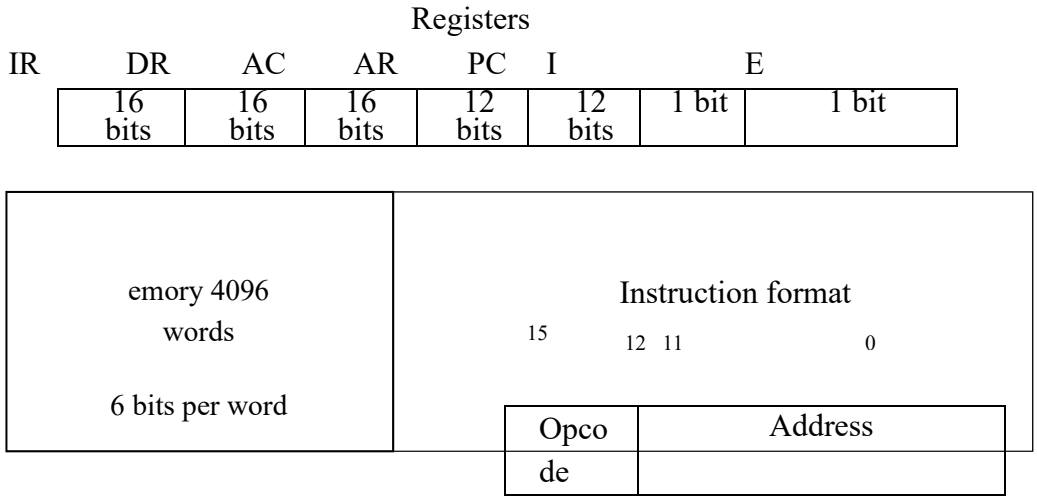
Memory hierarchy (main, cache and auxiliary memory), Input-Output Interface, Modes of Transfer: Programmed I/O, Interrupt initiated I/O, Direct memory access.

Practical **(30 hours)**

List of Practicals:

(Use Simulator – CPU Sim 3.6.9 or any higher version for the implementation)

1. Create a machine based on the following architecture:



Basic Computer Instructions

Memory Reference			Register Reference	
Symbol	Hex		Symbol	Hex
AND	0xxx	Direct Addressing	CLA	7800
ADD	1xxx		CLE	7400
LDA	2xxx		CMA	7200
STA	3xxx		CME	7100
BUN	4xxx		CIR	7080
BSA	5xxx		CIL	7040
ISZ	6xxx		INC	7020
AND_I	8xxx	Indirect Addressing	SPA	7010
ADD_I	9xxx		SNA	7008
LDA_I	Axxx		SZA	7004
STA_I	Bxxx		SZE	7002
BUN_I	Cxxx		HLT	7001
BSA_I	Dxxx		INP	F800
ISZ_I	Exxx		OUT	F400

Refer to Chapter-5 of reference 1 for description of instructions.

Design the register set, memory and the instruction set. Use this machine for the assignments of this section.

2. Create a Fetch routine of the instruction cycle.

3. Write an assembly program to simulate ADD operation on two user-entered numbers.
4. Write an assembly program to simulate SUBTRACT operation on two user-entered numbers.
5. Write an assembly program to simulate the following logical operations on two user-entered numbers.
    i. AND
    ii. OR
    iii. NOT
    iv. XOR
    v. NOR
    vi. NAND
6. Write an assembly program for simulating following memory-reference instructions.
    i. ADD
    ii. LDA
    iii. STA
    iv. BUN
    v. ISZ
7. Write an assembly language program to simulate the machine for following register reference instructions and determine the contents of AC, E, PC, AR and IR registers in decimal after the execution:
    i. CLA
    ii. CMA
    iii. CME
    iv. HLT
8. Write an assembly language program to simulate the machine for following register reference instructions and determine the contents of AC, E, PC, AR and IR registers in decimal after the execution:
    i. INC
    ii. SPA
    iii. SNA
    iv. SZE

9. Write an assembly language program to simulate the machine for following register reference instructions and determine the contents of AC, E, PC, AR and IR registers in decimal after the execution:

    i. CIR

    ii. CIL

10. Write an assembly program that reads in integers and adds them together; until a negative non-zero number is read in. Then it outputs the sum (not including the last number).

11. Write an assembly program that reads in integers and adds them together; until zero is read in. Then it outputs the sum.

Essential Readings

- David A. Patterson and John L. Hennessy. "Computer Organization and Design: The Hardware/Software interface", 5th edition, Elsevier, 2012.
- Mano, M. Computer System Architecture, 3rd edition, Pearson Education, 1993.

Suggestive Readings

- Mano, M. Digital Design, Pearson Education Asia, 1995.
- Null, L., & Lobur, J. The Essentials of Computer Organization and Architecture. 5th edition, (Reprint) Jones and Bartlett Learning, 2018.
- Stallings, W. Computer Organization and Architecture Designing for Performance 8th edition, Prentice Hall of India, 2010.

Note: Examination scheme and mode shall be as prescribed by the Examination Branch, University of Delhi, from time to time.

DISCIPLINE SPECIFIC CORE COURSE – 3: MATHEMATICS FOR COMPUTING

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course title & Code	Credits	Credit distribution of the course				Eligibility criteria	Pre-requisite c the course (if any)
		Lecture	Tutorial	Practical/ Practice			
Mathematics for Computing	4	3	0	1		Class XII pass	NIL

Learning Objectives

The Learning Objectives of this course are as follows:

- Introduces the students to the fundamental concepts and topics of linear algebra and vector calculus.
- To build the foundation for some of the core courses in later semesters.

Learning Outcomes

This course will enable the students to:

- Perform operations on matrices and sparse matrices.
- Compute the determinant, rank and eigenvalues of a matrix.
- Perform diagonalization.
- Perform operations on vectors, the dot product and cross product.
- Represent vectors geometrically and calculate the gradient, divergence, curl.
- Apply linear algebra and vector calculus to solve problems in sub-disciplines of computer science.

SYLLABUS OF DSC – 3

Theory

Unit – 1 **(6 hours)**

Introduction to Matrix Algebra

Echelon form of a Matrix, Rank of a Matrix, Determinant and Inverse of a matrix, Solution of System of Homogeneous & Non-Homogeneous Equations: Gauss elimination and Solution of System of Homogeneous Equations: Gauss Jordan Method.

Unit – 2 **(21 hours)**

Vector Space and Linear Transformation

Vector Space, Sub-spaces, Linear Combinations, Linear Span, Convex Sets, Linear Independence/Dependence, Basis & Dimension, Linear transformation on finite dimensional vector spaces, Inner Product Space, Schwarz Inequality, Orthonormal Basis, Gram-Schmidt Orthogonalization Process.

Unit – 3 **(9 hours)**

EigenValue and EigenVector

Characteristic Polynomial, Cayley Hamilton Theorem, Eigen Value and Eigen Vector of a matrix, Eigenspaces, Diagonalization, Positive Definite Matrices, Applications to Markov Matrices.

Unit – 4 **(9 hours)**

Vector Calculus

Vector Algebra, Laws of Vector Algebra, Dot Product, Cross Product, Vector and Scalar Fields, Ordinary Derivative of Vectors, Space Curves, Partial Derivatives, Del Operator, Gradient of a Scalar Field, Directional Derivative, Gradient of Matrices, Divergence of a Vector Field, Laplacian Operator, Curl of a Vector Field.

Practical **(30 hours)**

List of Practicals:

1. Create and transform vectors and matrices (the transpose vector (matrix) conjugate transpose of a vector (matrix))
2. Generate the matrix into echelon form and find its rank.
3. Find cofactors, determinant, adjoint and inverse of a matrix.
4. Solve a system of Homogeneous and non-homogeneous equations using Gauss elimination method.
5. Solve a system of Homogeneous equations using the Gauss Jordan method.
6. Generate basis of column space, null space, row space and left null space of a matrix space.
7. Check the linear dependence of vectors. Generate a linear combination of given vectors of Rn/ matrices of the same size and find the transition matrix of given matrix space.
8. Find the orthonormal basis of a given vector space using the Gram-Schmidt orthogonalization process.
9. Check the diagonalizable property of matrices and find the corresponding eigenvalue and verify the Cayley- Hamilton theorem.
10. Application of Linear algebra: Coding and decoding of messages using nonsingular matrices.
    eg code “Linear Algebra is fun” and then decode it.
11. Compute Gradient of a scalar field.
12. Compute Divergence of a vector field.
13. Compute Curl of a vector field.

Essential Reading

- Strang Gilbert. Introduction to Linear Algebra, 5th Edition, Wellesley-Cambridge Press, 2021.
- Kreyszig Erwin. Advanced Engineering Mathematics, 10th Edition, Wiley, 2015.
- Strang Gilbert. Linear Algebra and Learning from Data, 1st Edition, Wellesley-Cambridge Press, 2019.
- Jain R. K., Iyengar S.R. K. Advanced Engineering Mathematics, 5th Edition, Narosa, 2016.

Suggestive Reading

- Deisenroth, Marc Peter, Faisal A. Aldo and Ong Cheng Soon. Mathematics for Machine Learning, 1st Edition, Cambridge University Press, 2020.
- (Lipschutz Seymour and Lipson Marc. Schaum's Outline of Linear Algebra, 6th Edition, McGraw Hill, 2017.

B.A (Prog) with Computer Science as Major

CATEGORY-II

DISCIPLINE SPECIFIC CORE COURSE – 1: INTRODUCTION TO PROGRAMMING USING C++

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
Introduction to Programming using C++	4	3	0	1	Class XII pass	Nil

Learning Objectives

This course is designed to:

- Introduce programming concepts using C++ to students.
- Develop structured as well as object-oriented programming skills using C++ programming language.
- Achieve competence amongst its students to develop correct and efficient C++ programs to solve problems spanning multiple disciplines.

Learning outcomes

On successful completion of the course, students will be able to:

- Write simple programs using built-in data types of C++.
- Implement arrays and user defined functions in C++.
- Solve problems spanning multiple disciplines using suitable programming constructs in C++.
- Solve problems spanning multiple disciplines using the concepts of object oriented programming in C++.

SYLLABUS OF DSC - 1

Theory

Unit – 1 **(3 hours)**

Introduction to C++

Need and characteristics of Object-Oriented Programming, Structure of a C++ Program (main () function, header files, output, input, comments), compile and execute a simple program

Unit – 2 **(9 hours)**

Data types and Expressions

Keywords, built in data types, variables and constants, naming convention, Input-Output statements, operators and their precedence, expressions, typecasting, library functions

Unit – 3 **(12 hours)**

Control Constructs in C++

Decision making using selection constructs, iteration using looping constructs.

Unit – 4 **(6 hours)**

Arrays, Pointers and User Defined Functions

Defining and initializing single and multi-dimensional arrays, user defined functions, passing arguments to functions, returning values from functions, inline functions, default arguments, introduction to pointers

Unit – 5 **(15 hours)**

Classes and Objects

Need and implementation of abstraction, encapsulation, inheritance and polymorphism, creating classes, objects as function arguments, modifiers and access control, constructors and destructors.

Practical **(30 hours)**

List of Practicals:

1. Write a program to find the largest of n natural numbers.
2. Write a program to find whether a given number is prime or not.
3. Write a program that takes a positive integer n and the produce n lines of output as shown:

   *

   * *

   * * *

   * * * *

   (for n = 4)

4. Write a menu driven program for following:
   a. to check whether a given number is odd or even.
   b. display a fibonacci series
   c. compute factorial of a number
5. Write a program to accept a number, reverse it and print the sum of its digits.
6. Write a program using functions to print the series and its sum:

   $1 + 1/2! + 1/3! + \ldots + 1/n!$

7. Write a program to perform the following operations on an input string
   a. Print length of the string

b. Find frequency of a character in the string
c. Print whether characters are in uppercase or lowercase
d. to check whether a given string is palindrome or not.

8. Write a program that will prompt the user for a list of 5 prices. Compute the average of the prices and find out all the prices that are higher than the calculated average.
9. Design a class named Vehicle, having registration number and year as its private members. Define a suitable constructor and a method to print the details of a vehicle. Write a C++ program to test the above class.
10. Inherit a class Car from the Vehicle class defined above. Add model to the Car class. Define a suitable constructor and a method to print the details of a car. Write a C++ program to test inheritance of this class.

Essential Readings

- E. Balaguruswamy, Object Oriented Programming with C++,7th edition, McGraw-Hill Education, 2017.
- 2. Robert Lafore, Object Oriented Programming in C++, 4th edition, SAMS Publishing, 2008.

Suggestive Reading

- D.S. Malik, C++ Programming: From Problem Analysis to Program Design, 6th edition, Cengage Learning, 2013.
- (ii) Herbert Schildt, C++: The Complete Reference, 4th Edition, McGraw Hill, 2003.

Note: Examination scheme and mode shall be as prescribed by the Examination Branch, University of Delhi, from time to time.

DISCIPLINE SPECIFIC CORE COURSE – 2: PROGRAMMING FUNDAMENTALS USING PYTHON

CREDIT DISTRIBUTION, ELIGIBILITY AND PRE-REQUISITES OF THE COURSE

Course title & Code	Credits	Credit distribution of the course			Eligibility criteria	Pre-requisite of the course (if any)
		Lecture	Tutorial	Practical/ Practice		
Programming Fundamentals Using Python	4	3	0	1	Class XII pass	Nil

Learning Objectives

This course is designed to:

- Introduce programming concepts using Python to students.

- Develop structured as well as object-oriented programming skills using Python.
- Achieve competence amongst its students to develop correct and efficient Python programs to solve problems spanning multiple disciplines.

Learning Outcomes

On successful completion of this course, a student will be able to:

- Write simple programs using built-in data types of Python.
- Implement arrays and user defined functions in Python.
- Solve problems spanning multiple disciplines using suitable programming constructs in Python.
- Solve problems spanning multiple disciplines using the concepts of object-oriented programming in Python.

SYLLABUS OF DSC - 2

Theory

Unit – 1 **(6 hours)**

Introduction to Python Programming

Problem solving strategies; Structure of a Python program; Syntax and semantics; Python interpreter/shell, indentation; Executing simple programs in Python.

Unit – 2 **(12 hours)**

Creating Python Programs

Identifiers and keywords; literals, numbers, and strings; Operators and expressions; Input and output statements; control structures (conditional statements, loop control statements, break, continue and pass), Errors and exception handling.

Unit – 3 **(9 hours)**

User Defined Functions

Defining functions, passing arguments and returning values, default arguments

Unit – 4 **(18 hours)**

Built-in Data Structures

Strings, Lists, Tuples, Sets, Dictionaries; their built-in functions, operators and operations

Practical **(30 hours)**

List of Practicals:

1. WAP to calculate total marks, percentage and grade of a student. Marks obtained in each of three subjects are to be input by the user. Assign grades according to the following criteria:

   Grade A : if Percentage >=80

   Grade B : if Percentage >=60 and Percentage <80

Grade C : if Percentage >=40 and Percentage <60

Grade D : if Percentage <=40

2. WAP to print factors of a given number.
3. WAP to add N natural numbers and display their sum.
4. WAP to print the following conversion table (use looping constructs):

Height (in Feet)	Height (in inches)
5.0 ft	60 inches
5.1 ft	61.2 inches
5.8 ft	69.6 inches
5.9 ft	70.8 inches
6.0 ft	72 inches

5. WAP that takes a positive integer n and the produce n lines of output as shown:

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

(for n =4)

6. Write a menu driven program using user defined functions to print the area of rectangle, square, circle and triangle by accepting suitable input from user.
7. Write a function that calculates factorial of a number n.
8. WAP to print the series and its sum: (use functions)

   1/1! + 1/2! + 1/3! …….1/n!
9. WAP to perform the following operations on an input string
   a. Print length of the string
   b. Find frequency of a character in the string
   c. Print whether characters are in uppercase or lowercase
10. WAP to create two lists: one of even numbers and another of odd numbers. The program should demonstrate the various operations and methods on lists.
11. WAP to create a dictionary where keys are numbers between 1 and 5 and the values are the cubes of the keys.
12. WAP to create a tuple t1 = (1,2,5,7,2,4). The program should perform the following:
    a. Print tuple in two lines, line 1 containing the first half of tuple and second line having the second half.
    b. Concatenate tuple t2 = (10,11) with t1.

Essential Readings

- Kamthane, A. N., & Kamthane, A.A. Programming and Problem Solving with Python, McGraw Hill Education, 2017.
- Balaguruswamy E. "Introduction to Computing and Problem Solving using Python",2nd edition, McGraw Hill Education, 2018.
- Taneja, S., Kumar, N. Python Programming- A modular Approach, Pearson Education India, 2018.

Suggestive Readings

- Guttag, J. V. Introduction to computation and programming using Python, MIT Press, 2018.
- (ii) Downey, A. B. Think Python–How to think like a Computer Scientist 2nd edition. O'Reilly 2015.

Note: Examination scheme and mode shall be as prescribed by the Examination Branch, University of Delhi, from time to time.