Syllabi of Core Courses

This section gives the detailed syllabus of the core courses. Each course describes the course objective, learning outcomes, the units and the reading material. The reading material has 2 -3 components: main resource(/s), additional text material, and online resources. Main resources are kept to a minimum possible and no more than 3. Additional resources and the online material may be used to enhance the knowledge of the subject.

DSC 01: Programming using Python

Course Objective

This course is designed as the first course that introduces programming concepts using Python to Computer Science students. The course focuses on the development of Python programming to solve problems of different domains. It also introduces the concept of object- oriented programming.

Course Learning Outcomes

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

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

Syllabus

Unit 1 Introduction to Programming: Problem solving strategies; Structure of a Python program; Syntax and semantics; Executing simple programs in Python.

Unit 2 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 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 Object Oriented Programming: Introduction to classes, objects and methods; Standard libraries.

Unit 5 File and exception handling: File handling through libraries; Errors and exception handling.

References

1. Taneja, S., Kumar, N. *Python Programming- A modular Approach*, 1st edition, Pearson Education India, 2018.
2. Balaguruswamy E. *Introduction to Computing and Problem Solving using Python,* 2nd edition, McGraw Hill Education, 2018.

Additional References

(i) Brown, Martin C. *Python: The Complete Reference*, 2nd edition, McGraw Hill Education, 2018.

(ii) Guttag, J.V. *Introduction to computation and programming using Python,* 2nd edition, MIT Press, 2016.

Suggested Practical List

1. WAP to find the roots of a quadratic equation

2. WAP to accept a number ‘n’ and
    a. Check if ’n’ is prime
    b. Generate all prime numbers till ‘n’
    c. 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

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

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

4. 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)
5. 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.
6. WAP to swap the first n characters of two strings.
7. 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.
8. 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

9. WAP to read a file and
    a. Print the total number of characters, words and lines in the file.
    b. Calculate the frequency of each character in the file. Use a variable of dictionary type to maintain the count.
    c. Print the words in reverse order.
    d. Copy even lines of the file to a file named 'File1' and odd lines to another file named 'File2'.
10. 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.
11. Write a function that prints a dictionary where the keys are numbers between 1 and 5 and the values are cubes of the keys.
12. 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
13. 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.

DSC 02: Computer System Architecture

Course Objective

This course introduces the students to the fundamental concepts of digital computer organization, design and architecture. It aims to 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.

Course Learning Outcomes

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

1. Design Combinational Circuits using basic building blocks. Simplify these circuits

using Boolean algebra and Karnaugh maps. Differentiate between combinational circuits and sequential circuits.

2. Represent data in binary form, convert numeric data between different number systems and perform arithmetic operations in binary.
3. Determine various stages of instruction cycle, pipelining and describe interrupts and their handling.
4. Explain how CPU communicates with memory and I/O devices and distinguish between different types of processors.
5. Simulate the design of a basic computer using a software tool.

Syllabus

Unit 1 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 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 Data Representation and Basic Computer Arithmetic: Number System, r and (r-1)'s Complements, data representation and arithmetic operations.

Unit 4 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 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 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.

References

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

Additional References

(i) Mano, M. *Digital Design*, Pearson Education Asia, 1995.

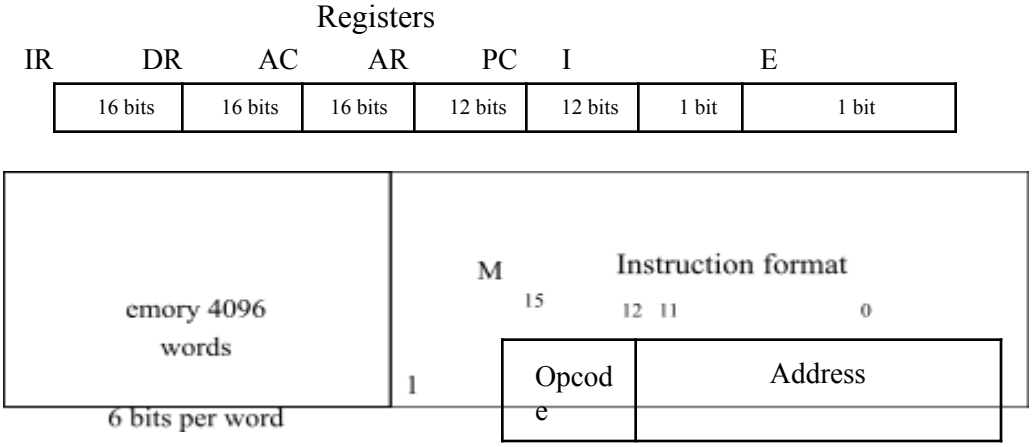
(ii) Null, L., & Lobur, J. *The Essentials of Computer Organization and Architecture*. 5th edition, (Reprint) Jones and Bartlett Learning, 2018.

(iii) Stallings, W. *Computer Organization and Architecture Designing for Performance* 8th edition, Prentice Hall of India, 2010.

Suggested Practical List

(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 Addressi	CLA	7800
ADD	1xxx		CLE	7400
LDA	2xxx		CMA	7200
STA	3xxx		CME	7100

BUN	4xxx	ng	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.

DSC 03: Mathematics for computing

Course Objective

This course introduces the students to the fundamental concepts and topics of linear algebra and vector calculus, whose knowledge is important in other computer science courses. The course aims to build the foundation for some of the core courses in later semesters.

Course Learning Outcomes

After successful completion of this course, the student will be able to:

1. Perform operations on matrices and sparse matrices
2. Compute the determinant, rank and eigenvalues of a matrix
3. Perform diagonalization

4. Perform operations on vectors, the dot product and cross product
5. Represent vectors geometrically and calculate the gradient, divergence, curl
6. Apply linear algebra and vector calculus to solve problems in sub-disciplines of computer science.

Syllabus

Unit 1 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 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 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 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.

References

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

2016.

Additional References

(i) Deisenroth, Marc Peter, Faisal A. Aldo and Ong ChengSoon. *Mathematics for Machine Learning*, 1st Edition, Cambridge University Press, 2020.

(ii) Lipschutz Seymour and Lipson Marc. *Schaum's Outline of Linear Algebra*, 6th Edition, McGraw Hill, 2017.

Suggested Practical List

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.