KRISHNA UNIVERSITY

BOARD OF STUDIES (BoS) – DEPARTMENT OF COMPUTER SCIENCE

MINUTES OF THE BOARD OF STUDIES MEETING

Agenda:

To discuss and approve the proposed **curriculum, course structure and syllabus for I and II Semesters of the following PG programmes**:

- **Computer Applications**
- **Artificial Intelligence and Robotics**
- **Artificial Intelligence and Machine Learning**
- **Cybercrime and Cyber Security**
- **Data Analytics**

Members Present:

S. No.	Name	Designation	Role
1	Prof. R. Kiran Kumar	Dept. of Computer Science, Krishna University	Chairman
2	Smt. P. Durga Bhavani	HOD, Dept. of Computer Science, Nalanda Degree College, Vijayawada	Member
3	Kondapalli Ramkumar	HOD, Dept. of Electronics and Internet of Things, Nalanda Degree College, Vijayawada	Member
4	Bommareddy Sudheer Reddy	Lecturer, Dept. of Computer Science, Nalanda Degree College & PG Centre, Vijayawada	Member
5	Prof. P. Suresh Varma	Dept. of CSE, Adikavi Nannaya University, Rajahmundry	Member
6	Dr. M.H.M. Krishna Prasad	Dept. of CSE, JNTUK, Kakinada	Member
7	Prof. D. Lalitha Bhaskari	Dept. of CS & Systems Engineering, Andhra University	Member
8	Assoc. Prof. M. Babu Reddy	HOD, Dept. of Computer Science, Krishna University	Member

Proceedings

The meeting commenced with the Chairman welcoming all the members. The proposed curriculum and syllabi for the **I and II Semesters of the PG programmes** were placed before the Board for discussion and consideration.

The members discussed the proposed **course structure, syllabus , credits, teaching scheme, practical components, Course Outcomes and assessment pattern** of the above programmes. The members suggested that the curriculum should be practical, application-oriented and aligned with current technological and industry requirements.

Resolution

After detailed discussion, the **Board of Studies approved/recommended the proposed curriculum and syllabus for the I and II Semesters** of the following PG programmes:

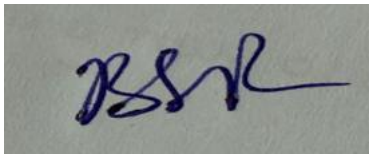
1. **Computer Applications**
2. **Artificial Intelligence and Robotics**
3. **Artificial Intelligence and Machine Learning**
4. **Cybercrime and Cyber Security**
5. **Data Analytics**

The Board recommended that the curriculum, syllabus, Course Outcomes, credits, teaching scheme and assessment pattern be finalized by incorporating the suggestions of the members and submitted to the competent academic authority for approval and implementation.

Conclusion

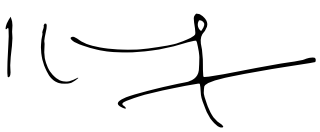
The Chairman thanked all the members for their valuable suggestions and active participation. The meeting concluded with a vote of thanks.











Model Syllabus for PG DIPLOMA in Computer Applications

COURSE STRUCTURE (for Semester I to VI)

Year	Semester	Course	Title of the Course	No. of Hrs /Week	No. of Credits
I	I	1	Data Structures using C	3	3
			Data Structures using C Practical	2	1
	II	2	Python Programming	3	3
			Python Programming-Practical	2	1

SEMESTER- I

COURSE 1: DATA STRUCTURES USING C

Theory **Credits: 3** **3 hrs/week**

Course Objectives:

1. Understand the fundamentals of computer programming, Apply structured problem-solving approaches using algorithms, flowcharts, and C programming constructs.
2. Develop efficient logic using decision-making, loop, and jump control statements.
3. Utilize derived data types like arrays and strings for modular program design.
4. Design and implement modular solutions using functions, recursive logic, pointer operations, and dynamic memory management.
5. Handle complex data structures including structures, unions, and text file operations.

Course Outcomes:

At the End of the Course, The Students will be able to:

1. Understand basic computing concepts, programming paradigms and write structured C programs.
2. Apply control flow statements to solve logical and repetitive tasks in C.
3. Implement arrays and string operations to manage and manipulate data efficiently.
4. Design modular code using functions, recursion, and appropriate parameter passing.
5. Utilize pointers and memory operations for effective data handling. Demonstrate competence in dynamic memory allocation and text file processing.

Unit 1. Basic Concepts:

Algorithm: Definition and characteristics, Complexity analysis: Space Complexity, Time Complexity, Asymptotic Notations.

Introduction to Data structures: Definition, Types of Data structures, Abstract Data Types (ADT), Introduction to Linked Lists, Representation of linked lists in Memory, Comparison between Linked List and Array.

Unit 2. Linked Lists:

Types of Linked Lists - Singly Linked list, Doubly Linked list, Circularly Singly Linked list, Circularly Doubly Linked list; Implementation of Single Linked List ADT: Creating a List, Traversing a linked list, Searching in linked list, Insertion and deletion into linked list (At first Node, Specified Position, Last node).

Unit 3. Stacks and Queues:

Introduction to stack ADT, Implementation of stacks using array and Linked List, Application of stacks - Polish Notations - Converting Infix to Post Fix Notation - Evaluation of Post Fix Notation.

Queues: Introduction to Queue ADT, Implementation of Queues using array and Linked List, Application of Queues Types of Queues- Circular Queues, De-queues, Priority Queue, Heaps.

Unit 4. Searching and Sorting:
Linear or Sequential Search, Binary Search, Hashing and collision resolution.

Sorting: Selection Sort, Bubble Sort, Insertion Sort, Quick Sort and Merge Sort

Unit 5. Trees and Graphs:
Tree Terminology, Binary Tree Representation, Traversal techniques, Expression Tree, Binary Search Tree- Definition, Operations on a Binary Search Tree: Creation, Search, Insertion & deletion.

Graphs: Introduction to Graphs, Terminology, Representation (Adjacency Matrix, Adjacency List), Traversal of Graphs (DFS, BFS), Applications of Graphs, Concept of Shortest Path Problems, Concept of Minimum Cost Spanning Tree

Textbooks:

1. Data Structures Using C, Balagurusamy E. Tata MCGraw Hill
2. Data Structures using C, Reema Thareja, Third Edition, Oxford University Press

Reference Books:

1. Data Structures, Lipschutz, Schaum's Outline Series, Tata Mcgraw-hill
2. Data Structures Using C, Ch. Vijay Kumar, Pen Press International

Activities:

Outcome: Explain algorithm characteristics, time and space complexity, and asymptotic notations with clarity

Activity: Create a comparative chart of algorithms with different notations related to time and space complexities.

Evaluation Method: Rubric-based assessment of the chart for correctness, clarity, and depth of explanation on a 10-point scale.

Outcome: Implement and analyze different types of linked lists, including insertion, deletion, and traversal operations

Activity: Code a menu-driven program in C to implement single linked lists with all basic operations.

Evaluation Method: Practical lab assessment with test cases and Viva-style questioning to explain pointer manipulation.

Outcome: Develop stack and queue data structures using arrays and linked lists, and apply them in expression evaluation

Activity: Build a program to convert infix expressions to postfix and evaluate them using stacks; Implement queues using both arrays and linked lists with enqueue/dequeue operations.

Evaluation Method: Code review and execution of programs for sample cases and evaluation based on correctness and efficiency.

Outcome: Apply efficient searching and sorting algorithms to solve computational problems and evaluate performance trade-offs

Activity: Implement and compare sorting algorithms (e.g., selection sort and bubble sort) and searching algorithms (e.g., Linear vs. Binary Search) on datasets of varying sizes. Record number of swaps and iterations for preparing a chart to assimilate the results.

Evaluation Method: Performance report with graphs and analysis. Oral presentation or peer review discussing trade-offs and algorithm selection rationale.

Outcome: Construct and traverse tree and graph structures, using them to solve problems like shortest path and spanning trees

Activity: Implement binary trees and graphs using adjacency lists/matrices.

Evaluation Method: Lab demo with sample inputs and visual output (e.g., tree traversal order, graph paths).

SEMESTER-I

COURSE 1: DATA STRUCTURES USING C

Practical **Credits: 1** **2 hrs/week**

List of Experiments

1. Write a program to read 'N' numbers of elements into an array and also perform the following operation on an array
   a. Add an element at the beginning of an array

   b. Insert an element at given index of array

   c. Update an element using a values and index

   d. Delete an existing element

2. Write a program to implement Single Linked List with insertion, deletion and traversal operations
3. Write a program to implement Doubly Linked List with insertion, deletion and traversal operations
4. Write a program to implement the Stack operations using Arrays and Linked Lists.
5. Write a program to convert a given infix expression to a postfix expression using stacks.
6. Write a program to implement the Queue operations using Arrays and Linked Lists.
7. Write a program to implement the Circular Queue operations using Arrays.
8. Write a program for Binary Search Tree Traversals
9. Write a program to search an item in a given list using the following Searching Algorithms
   a. Linear Search

   b. Binary Search.

10. Write a program for implementation of the following Sorting Algorithms
    a. Bubble Sort

    b. Insertion Sort

    c. Quick Sort

    d. Merge Sort

SEMESTER-II

COURSE 2: PYTHON PROGRAMMING

Theory **Credits: 3** **3 hrs/week**

Course Objectives

1. **Introduce the foundational concepts** of Python programming including its syntax, IDEs, and control structures.
2. **Develop proficiency in modular programming** using functions, lambda expressions, recursion, and Python's built-in modules and packages.
3. **Explore core data structures** like strings, lists, tuples, and dictionaries for effective data manipulation.
4. **Teach exception handling mechanisms** and the use of regular expressions for pattern matching and text processing.
5. **Enable students to interact with files and databases** using Python to build real-world applications involving persistent storage and data retrieval.

Course Outcomes

At the end of the Course, the students will be able to

1. **Write and execute structured Python programs** using variables, expressions, and flow control statements.
2. **Implement modular code** leveraging functions, argument types, recursion, and reusable libraries.
3. **Manipulate and organize data efficiently** using Python's string operations and complex data structures.
4. **Handle runtime errors and apply regular expressions** for robust and flexible program behaviour.
5. **Perform file operations and connect to databases** through Python scripts to store, retrieve, and manage data effectively.

Unit 1. Basics of Python Programming:

Features of python, history of python, Python IDEs, Writing and Executing Python Program, literal constants, variables and identifiers, Data types, input operation- comments, Reserved words, Indentation, Operators and Expressions: Expressions in python, Operations in Strings, Other Data types, Type conversion, Decision control Statements, Iterative Statements, Nested loops, break, Continue, Pass Statements, else statement used with loops.

Unit 2. Strings and Collections:

Strings: Built-in String Methods and Functions

Lists: Access values in List, Updating values in Lists, Nested lists, Basic list operations, List Methods.

Tuple: Creating, Accessing, Updating and Deleting Elements in a tuple,Nested tuples.

Dictionaries: Creating a dictionary, Accessing values, Modifying an Entry, Deleting items, Built-in Dictionary Functions and Methods

Unit 3. Functions and Modules:
Function Definition, Function Call, Variable Scope and lifetime, The return statement, Required Arguments, Keyword Arguments, Default Arguments and Variable Length Arguments, Lambda Functions, Recursive Functions.

Importing Libraries, Modules, Packages in python, Standard library modules- Globals(), Locals(), and Reload(), Function Redefinition.

Unit 4. Exception Handling:
Errors and Exceptions, Handling Exceptions, Multiple Except blocks, Multiple Exceptions in a single block, Except Block without Exception, The else clause, Built-in and user-defined Exceptions, The finally block, Re-raising Exception, Assertions in python

Unit 5. File Handling & Database Connectivity:
Types of files in Python, Opening and Closing files, Reading and Writing files: write() and writelines() methods- append() method, read() and readlines() methods, Splitting words, File Positions.

Database connectivity using Python, Executing SQL commands using python, Assimilating SQL command results using python.

Textbooks:

1. Python Programming using Problem Solving Approach Reema Thareja Oxford University Press 2020
2. Exploring Python, Budd T A, McGraw-Hill Education, 1st Edition, 2011.

Reference Book:

1. Python: The Complete Reference, Martin C. Brown, Mc Graw Hill, 2018
2. Fundamentals of Python, Kenneth A. Lambert. (2019), First Programs,2nd Edition, CENGAGE Publication.

Activities:

Outcome: Write and execute structured Python programs using variables, expressions, and flow control statements.

Activity: Create a calculator program that uses variables, arithmetic expressions, and flow control (if-else) to perform basic operations (add, subtract, multiply, divide) based on user input.

Evaluation Method: Code walkthrough and output validation. Use a checklist to assess on a 10-point scale to check the:

- Correct use of variables and expressions
- Proper flow control logic
- Accurate results for different inputs

Outcome: Implement modular code leveraging functions, argument types, recursion, and reusable libraries.

Activity: Build a factorial calculator using both iterative and recursive functions. Include parameterized functions and import the math library for comparison.

Evaluation Method: Code review and oral explanation. Assess on 10-point scale based on:

- Function structure and argument usage
- Recursion logic
- Use of reusable libraries

Outcome: Manipulate and organize data efficiently using Python's string operations and complex data structures.

Activity: Develop a contact manager that stores names and phone numbers using dictionaries and lists. Include string formatting and search functionality.

Evaluation Method: Practical demo with test cases. Evaluate based on:

- Use of string methods (split, join, format)
- Data structure selection and manipulation
- Search and retrieval accuracy

Outcome: Handle runtime errors and apply regular expressions for robust and flexible program behaviour.

Activity: Create a form validator that checks email and phone number formats using regular expressions. Include try-except blocks to handle invalid inputs.

Evaluation Method: Scenario-based testing. Assess based on:

- Regex accuracy for pattern matching
- Robust error handling
- Program stability with edge cases

Outcome: Perform file operations and connect to databases through Python scripts to store, retrieve, and manage data effectively.

Activity: Build a student grade logger that reads from a CSV file, stores data in a SQLite database, and allows querying by student name.

Evaluation Method: Lab test with sample data. Evaluate on a 10-point scale:

- File read/write operations
- Database connection and query execution
- Data integrity and retrieval accuracy

SEMESTER-II

COURSE 2: PYTHON PROGRAMMING

Practical **Credits: 1** **2 hrs/week**

List of Experiments

1. Display a welcome message using print().
2. Declare and use identifiers belonging to strings, integers, floats, and booleans.
3. Accept user input (name, age, height, student status) and display each value with its type using type().
4. Perform operations like .upper(), .find(), .replace() on strings.
5. Write a program to reverse the string, count vowels and words.
6. Write a program for slicing, sorting, and list comprehension.
7. Program to apply list methods: append(), extend(), insert(), remove(), pop(), sort().
8. Create tuples to store student (name, age, course) data and perform
    a. Accessing elements using indexing and slicing.
    b. Demonstrate immutability by attempting to modify a tuple.
    c. Create and navigate nested tuples.
9. Create a dictionary with student roll number as keys and names/marks as values.
    a. Accessing, adding, updating, and deleting key-value pairs.
    b. Iterating through keys, values, and items.
10. Write a program to demonstrate variable length arguments.
11. Write a program to illustrate lambda and recursive functions.
12. Write a program to demonstrate globals(), locals(), and reload() functions.
13. Demonstrate exception handling and assertions in Python.
14. Write a Python program to copy the contents of one file into another in reverse order.
15. Working with Datasets using Numpy and Pandas
16. Write a program to connect to the database and retrieve the required information using SQL commands.

KRISHNA UNIVERSITY

PG DIPLOMA Model Paper for Core Subjects

Semester :I

Paper Title : DataStructures using 'C'

Time : 3hrs **Max. Marks:70**

Section –A(20 Marks)

Answer any *Five* questions. All questions carry equal marks. 5X4=20M

1. Define ADT
2. Explain Space Complexity
3. Doubly Linked list,
4. What is Polish Notations
5. Application of Queues
6. What is Bubble Sort?
7. What is BST?
8. Explain Minimum Cost Spanning Tree

Section–B (50Marks)

Answer ALL Questions. All questions carry equal marks. 5 X 10 = 50M

9. a) Explain the major applications of NLP

OR

b) Explain Array operations?

10. a) Define linkedlist. Exaplin types of Linked list with Example

OR

B). Discuss Opeartions on Linked list

11. a) Define Stack.Explain operations on Stack.

OR

B). Define Queue . Explain types of Queues with neat representation

12. a) Discuss Binary Search.

OR

b) Explain Merge Sort with example.

13. a) Define Tree. Explain Binary tree with Example

OR

b) Explain Traversal of Graphs

KRISHNA UNIVERSITY

PG DIPLOMA Model Paper for Core Subjects

Semester :II

Paper Title : Python Programming

Time : 3hrs **Max. Marks:70**

Section –A(20 Marks)

Answer any *Five* questions. All questions carry equal marks. 5X4=20M

1. Explain objects of python programming
2. What is Pass Statement
3. What is dictionary?
4. What is Recursive Function?
5. Explain Packages in python.
6. Define Assertions
7. What is append() method
8. Write a short notes of SQL commands

Section–B (50Marks)

Answer ALL Questions. All questions carry equal marks. 5 X 10 = 50M

9. a) Explain Sequences in python with Example.

OR

b) Explain control Statements with Exmples

10. a) Explain String Functions with example.

OR

b) define Tuple. Explain tuple operations with example.

11. a) Explain Buit- in - functions with suitable example.

OR

b) Discuuss about Modules.

12. a) What is Exception ? How to avoid run time errors in a java program

OR

b) Explain User defined exceptions with example.

13. a) Define File. Explain various file operations in python

OR

b) Discuss about DBAPI