Problem Solving using Computers
(BSCS01)
Core Course - (CC) Credit:6

Course Objective

This course is designed as the first course in programming to develop problem solving skills. The course focuses on modularity, reusability, code documentation, and debugging skills. It also introduces the concept of object-oriented programming.

Course Learning Outcomes

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

1. describe the components of a computer and the notion of an algorithm.
2. apply suitable programming constructs and data structures to solve a problem.
3. develop, document, and debug modular python programs.
4. use classes and objects in application programs.
5. use files for I/O operations.

Unit 1

Computer Fundamentals and Problem Solving: Basic Computer Organization: CPU, memory, I/O Units. Problem solving using computer, notion of an algorithm.

Unit 2

Introduction to Python Programming: Python interpreter, using python as calculator, python shell, indentation, identifiers and keywords, literals, strings, arithmetic, relational and logical operators.

Unit 3

Creating Python Programs: Input and output statements, defining functions, control statements default arguments, errors and exceptions.

Unit 4

Inbuilt Data Structures: strings, lists, sets, tuples, nested lists, built-in functions, dictionary and associated operation.

Unit 5

Object Oriented Programming: Introduction to Classes, Objects and Methods, Standard Libraries, File handling through libraries.

Unit 6

Sorting and Searching: Iterative and Recursive methods for searching and sorting

Practical

1. Execution of expressions involving arithmetic, relational, logical, and bitwise operators

in the shell window of Python IDLE.

2. Write a Python function to produce the following outputs.

(a)
```
*
**
***
*****
******
*******
```

(b)
```
$ $ $ $ $
$ $
$ $
$ $
$ $ $ $ $
```

3. Write a Python program to illustrate the functions of math module specified by the instructor in laboratory.
4. Write a Python program to produce a table of sins, cosines and tangents. Make a variable x in range from 0 to 10 in steps of 0.2. For each value of x, print the value of sin(x), cos(x) and tan(x).
5. Write a menu driven program to calculate the area of Square, rectangle, circle and triangle. Use suitable assertions.
6. Write a Python function that takes a number as an input from the user and computes its factorial. Then find the sum of the n terms of the following series:

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

7. Write a Python function to return n th terms of Fibonacci sequence
8. Write a function that takes a number as an input and finds its reverse and computes the sum of its digits.
9. Write a function that takes two numbers as input parameters and returns their least common multiple.
10. Write a function that takes a number as an input and determine whether it is prime or not.
11. Write a Python function that takes a string as an input from the user and displays its reverse.

More exercises using sets, lists and dictionary, as announced by instructor in the laboratory.

References

1. Downey, A.B. (2008). *Think Python–How to think like a Computer Scientist*. Needham, Massachusetts : Green Tea Press.
2. Urban, M. & Murach, J. (2018). *Python Programming*. Shroff.

Additional Resources

1. Guttag, J. V. (2013). *Introduction to computation and programming using Python.* MIT Press.
2. Liang, Y. D. (2013). *Introduction to Programming using Python.* Pearson.
3. Taneja, S. & Kumar, N. (2018). *Python Programming - A modular Approach.* Pearson.

Teaching Learning Process

- Talk and chalk method
- Computer based presentations by teachers.
- Group Discussions
- Assignments
- Offline and online Quiz
- Presentations by group of students for enhanced learning.

Tentative weekly teaching plan is as follows:

Week 1 - 2	**Unit 1-Computer Fundamentals and Problem Solving**: Basic Computer Organization: CPU, memory, I/O Units. Problem solving using computer, notion of an algorithm.
Week 3 - 4	**Unit 2-Introduction to Python Programming:** Python interpreter, using python as calculator, python shell, indentation. identifiers and keywords, literals, strings, arithmetic, relational and logical operators.
Week 5 - 7	**Unit 3- Creating Python Programs:** input and output statements, defining functions, control statements, default arguments, errors and exceptions.
Week 7 - 9	**Unit 4 - Inbuilt Data Structures:** Strings, lists, sets, tuples, nested lists, built-in functions, dictionary and associated operation.
Week 10 - 11	**Unit 5 - Object Oriented Programming:** Introduction to Classes, Objects and Methods, Standard Libraries, File handling through libraries.

Week 12 - 15	**Unit 6 - Sorting and Searching:** Iterative and Recursive methods for searching and sorting

Assessment Methods

- Unit-wise assignments, presentations, viva, quiz,as announced by the instructor in the class
- End semester exam
- Internal assessment

Keywords

Problem Solving, Control structure, Functions, Strings, Lists, Object Oriented Programming

---

Database Management Systems
(BSCS02)
Core Course –(CC) Credit:6

Course Objective

The course introduces the students to the fundamentals of database management systems and methods to store and retrieve data. The course would give students hands-on practice of structured query language in a relational database management system.

Course Learning Outcomes

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

1. use database management system to manage data.
2. create entity relationship diagrams for modeling real-life situations and design the database schema.
3. use the concept of functional dependencies to remove data anomalies and arrive at normalized database design.
4. write queries using relational algebra and SQL.

Unit 1

Introduction to DBMS: Introduction to Database Management Systems, characteristics of database approach, data models, DBMS architecture and data independence.

Unit 2