1. Introduction to Generic Electives (Computer Science Department)

Generic elective of Computer Science and Applications are designed to impart flavour of computer science and computer applications to the students who do not have an opportunity to study these subject either major or minor subjects in their curriculum. The courses are designed for students studying BA/BSc (Hons) programme, and those studying BA/BCom. (Non-Hons) programme of University of Delhi.

---

2. Credit Distribution for Generic Electives (Computer Science Department)

All Generic Elective courses are 6 credit courses with breakup of 4 credits for theory to be taught in 4 hours per week, and 2 credits for 4 hours of laboratory work per week.

---

3. Semester-wise Distribution of Courses.

As per the UGC curriculum for undergraduate courses, BA/BSc (Hons) programme students study generic electives in Semester **I, II, III** and **IV**. The semester is amply reflected in the course code. The first number in the course code indicates the semester in which the course will be offered. For example, course CSGE**1**XX is offered in Semester **I**, and is for students studying BA/BSc (Hons) programme.

BA/BCom (Non-Hons) programme students study generic electives in Semester **V** and **VI**. The first number in the course code indicates the semester in which the course will be offered. For example, course CSGE**5**XX is offered in Semester **V** for those studying BA/ BCom (Non-Hons) programme.

---

4. Generic Elective Courses: Computer Science for Hons Courses

---

Programming using Python (CSGE101) Generic Elective - (GE)

Credit: 06

Course Objective

This course is designed as the first course that introduces computers and programming to non-Computer Science students. The course focuses on the use of computer and 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. Describe the components of a computer and notion of an algorithm.
2. Apply suitable programming constructs and built-in data structures to solve a problem.
3. Develop, document, and debug modular python programs.
4. Use classes and objects in application programs and visualize data.

Detailed Syllabus

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/shell, indentation; identifiers and keywords; literals, numbers, and strings; operators (arithmetic operator, relational operator, Boolean operator, assignment, operator, ternary operator and bitwise operator) and expressions

Unit 3

Creating Python Programs: Input and output statements, defining functions, control statements (conditional statements, loop control statements, break, continue and pass, exit function.), default arguments, errors and exceptions.

Unit 4

Strings and Lists: String class, built-in functions for string, string traversal, string operators and operations; Lists creation, traversal, slicing and splitting operations, passing list to a function

Unit 5

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

Unit 6

Built-in data structures: Tuples, sets, dictionary, stacks, and queues; searching and sorting.

Practicals

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 outputs such as:

(a)
```
 1
21
321
4321
```

b)
```
   1
  121
 12321
1234321
 12321
  121
   1
```

3. Write a Python program to illustrate the various functions of math module.
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 program that reads an integer value and prints “leap year” or “not a leap year”
6. Write a menu driven program to calculate the area of given building. Accept suitable inputs and use suitable assertions.
7. Write a Python function that takes a number as an input from the user and computes its factorial.
8. Write a Python function to generate the Fibonacci sequence till a given number “n”.
9. Write a function that takes a number as an input and finds its reverse and computes the sum of its digits.
10. Write a function that takes two numbers as input parameters and returns their least common multiple.

11. Write a function that takes a number as an input and determine whether it is prime or not.
12. Write a function that finds the sum of the
    a) first n odd terms
    b) first n even terms
    c) 1, 2, 4, 3, 5, 7, 9, 6, 8, 10, 11, 13.. till n-th term
13. Write a Python function that takes a string as an input from the user and determines whether it is palindrome or not.
14. Write a function that takes a sentence as input from the user and calculates the frequency of each letter. Use a variable of dictionary type to maintain the count.
15. Write a Python 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). Write a program to perform following operations:
    a. Print half the values of 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) witht1.
    d. Return maximum and minimum value from this tuple.
17. Write a function called "check_duplicates" that takes a list and returns true if there is any element that appears more than once. Also find the frequency of that element. The original list should not be modified.
18. Write a program to implement a class for finding area and perimeter of a rectangle. Write constructor, destructor, and functions for calculating area and perimeter.
19. Write a menu driven program to perform the following functions on strings:
    a. Find the length of string
    b. Return maximum of three strings
    c. Accept a string and replace every successive character with '#' Example-For Given string 'Hello World' returned string is 'H#l#o W#r#d'.
    d. Find number of words in the given string
20. Write a Python program to perform the following using list:
    a. Check if all elements in list are numbers or not
    b. If it is a numeric list, then count number of odd values in it
    c. If list contains all Strings, then display largest String in the list

    d. Display list in reverse form
    e. Find a specified element in list
    f. Remove the specified element

21. Usage of Python debugger tool-pydb and PythonTutor.
22. Implementation of Linear and binary search techniques
23. Implementation of selection sort, insertion sort, and bubble sort techniques

References

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

2. Taneja, S., Kumar, N. (2018). *Python Programming- A modular Approach*. Pearson Education India.

Additional Resources

1. Kamthane, A. N., & Kamthane, A.A. (2017) *Programming and Problem Solving with Python*, McGraw Hill Education.

2. Liang, Y. D. (2013). *Introduction to Programming using Python*. Pearson Education.

Course Teaching Learning Process

- Use of ICT tools in conjunction with traditional class-room teaching methods
- Interactive sessions
- Class discussions

Tentative weekly teaching plan is as follows:

Week	Topics
1-2	Computer Fundamentals and Problem Solving: Basic Computer Organization: CPU, memory, I/O Units, Problem solving using computer, notion of an algorithm
3	Python interpreter/shell, indentation; identifiers and keywords; Creating Python Programs:Input and output statements, defining functions, literals, numbers, and strings;
4	Operators: arithmetic operators, relational operators, boolean operators, assignment operators, ternary operator and bitwise

	operator and expressions
5	Control statements (conditional statements, loop control statements,
6	Break, continue and pass, exit function, default arguments
7	Errors and exceptions
8	Strings and Lists: String class, built-in functions for string, string traversal, string operators and operations;
9	Lists creation, traversal, slicing and splitting operations, passing list to a function
10	Object Oriented Programming: Introduction to Classes, Objects and Methods,
11	Standard Libraries, File handling through libraries
12-13	Built-in data structures: Tuples, sets, dictionary, stacks, and queues
14-15	searching and sorting

Assessment Methods

Written tests, assignments, quizzes, presentations as announced by the instructor in the class.

Keywords

Computer Hardware Organization, Problem solving for computer programming, Object oriented programming, Python

Database Management System (CSGE201) Generic Elective - (GE)

Credit: 06

Course Objectives

The course introduces the concepts of database management systems to students, focusing on basics such as the importance and significance of a database, data model, schema creation and normalization.

Course Learning Outcomes