# CS101T and CS101P: Data Structures using C

**Objective:**

The objective of the course is to introduce the fundamentals of C programming language and develop the skills for solving problems using computers. After completion of this course, a student will be able to

- Understand and use the process of abstraction using a programming language such as 'C'
- Analyse step by step and develop a program to solve real world problems
- Understand File handling in C and the basics of Graphics programming.
- Understanding and implementing Linear and Nonlinear data structures: Arrays, lists, stacks, queues, trees and graphs
- Implementation of various sorting and searching methods and their complexity analysis.

**Outline of the Course**

| Minimum Class Hours | | | Exam time (Hours) | | Marks | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Theory | Practical | Total | Theory | Practical | Theory | | Practical | | Total |
| | | | | | External | Internal | External | Internal | |
| 60 | 40 | 100 | 2 | 3 | 37.5 | 12.5 | 37.5 | 12.5 | 100 |

| Unit | Topic | Minimum Class Hours | | | Marks (Theory) |
|---|---|---|---|---|---|
| | | Theory | Practical | Total | |
| I | C fundamentals, I/O functions, Control statements, Functions, Arrays and Pointers, Structure and Union | 15 | 10 | 25 | 9.5 |
| II | Linear Data Structures: Linked List, Stacks and Queues | 10 | 10 | 20 | 6.5 |
| III | Trees | 15 | 7 | 22 | 8.5 |
| IV | Graphs | 10 | 8 | 18 | 6.5 |
| V | Searching and Sorting, and their complexity analysis | 10 | 5 | 15 | 6.5 |
| **Total** | | 60 | 40 | 100 | 37.5 |

## Detailed Syllabus

**Unit I: C Fundamentals, I/O functions, Control statements, Functions, Arrays and Pointers, Structure and Union** **15 Hours**

*C Fundamentals:* Algorithms, Flow charts, Development of algorithms, The C character set, identifiers and keywords, Data types, constants, variables and arrays, declarations, symbolic constants, Operators (Arithmetic, unary, relational, logical, bitwise, assignment),

*I/O functions:* Header files (stdio.h, conio.h) getch(), getche(), getchar(), putch(), putchar(), scanf(), printf(), gets(), puts(), clrscr(), window().

*Control statements:* Decision making and branching (**if..else**, **switch)**, Decision making and looping (**while**, **do .. while**, **for**), Jumping (**break**, **continue**, **goto),** Nested loops.

*Functions:* Overview (definition, declaration), defining and accessing a function, function prototypes, call by value, call by reference, recursion, Advantages and disadvantages of recursion over iteration, Storage classes (Automatic, Register, External, Static), String functions , Math functions , Memory allocation functions,

*Arrays and Pointers:* Defining an array, array initialization, processing an array, passing array to a function, multidimensional arrays, arrays and strings, pointer declarations, passing pointer to a function, pointer and one dimensional arrays, Operation on pointers.

*Structures and Unions:* Defining a structure, processing a structure, user defined data types, structures and arrays, structures and pointers, passing structures to a function, self-referential structures, Union, Union of structures, Enumerated, typedef.

**Unit II: Linear Data Structures: Linked List, Stacks and Queues** **10 Hours**

Data Type, Abstract Data Type, Data Structure, Complexity measures in terms of time and space; Big O notation. Linked List as a data structure (characteristics, advantages, disadvantages); operations on lists (creation, insertion, deletion, traversal, merging, splitting); singly linked list, doubly linked list, circular list; use of linked lists for polynomial representation and manipulation (addition and multiplication).

Stacks and Queues as data structures; implementation of stacks and queues using arrays and linked lists; Definitions of Circular Queue, Priority Queue, D-Queue; Application of stacks : Conversion of infix(containing arithmetic operators including exponential operator, and parenthesis) to postfix, evaluation of postfix expression.

**Unit III: Trees** **15 Hours**

Definition of tree as a data structure (Binary Trees and General Trees), Basic Terms (father, son, descendant, ancestor, height, depth, leaf, node, forest, ordered trees, strictly binary tree, complete binary tree, internal nodes, external nodes); Representation of trees using linked lists, Binary tree traversal methods (pre-order, in-order, post-order), recursive algorithms for traversal methods, Binary search trees (creation, insertion and deletion of a node), Height balanced (AVL) binary trees (construct and traverse an AVL tree), Definitions and characteristics of threaded binary trees, multi-way search trees and B-tree.

**Unit IV: Graphs**

Definition of a graph, Basic Terms, Graph representation: Adjacency matrix, adjacency lists, incidence matrix, adjacency multi-lists; Traversal schemes: Depth first search, Breadth first search (Recursive and non-recursive algorithms); Shortest Path algorithms (Dijkstra's), Spanning tree, Minimal spanning tree algorithms (Kruskal's algorithm)

**Unit V: Searching and Sorting, and their complexity analysis** **10 Hours**

Linear and binary search and their complexity analysis; Hashing, Hash Functions (division method, mid square method, folding), Analysis of ideal hash function; Conflict resolution (linear and quadratic probe, double hashing, separate chaining, coalesced chaining); Analysis of collision resolution techniques; Sorting algorithms(Insertion, Selection, Bubble, Quick) and comparison of their time complexity.

**Instructions For Paper Setter**

The question papers will be set according to the following scheme

*Cont'd ...*

| **Unit** | **Theory Questions** | | **Practical Questions** | | |
|---|---|---|---|---|---|
| | **To be set** | **To be answered** | **To be set** | **To be answered** | **Marks** |
| I | 2 | 1 | | | |
| II | 2 | 1 | 2 | 1 | 20 |
| III | 2 | 1 | | | |
| IV | 2 | 1 | 2 | 1 | 10 |
| V | 2 | 1 | 2 | 1 | 37.5 |

**Distribution of marks for Practical**

| | |
|---|---|
| 10% | : Syntax and input/output screens |
| 30% | : Logic and efficiency (source code, pseudocode, and algorithm) |
| 20% | : Error trapping (illegal or invalid input, stack overflow, underflow, insufficient physical memory etc.) |
| 20% | : Completion |
| 20% | : Result |

## Recommended Books

***Textbooks*:**

1. **Yashavant Kanetkar**, *Let us C* , BPB Publication
2. **S. Chattopadhyay, D. Ghosh Dastidar, M Chattopdhyay**, *Data Structures Through C Language*, BPB Publications, 2001
3. **William M. Newman, Robert F. Sproull**, *Principles of Interactive Computer Graphics*, Tata McGraw Hill Publishing Co Ltd.,

***Reference books:***

1. **Byron S. Gottfried**, *Theory and Problems of Programming with C*, Tata McGraw Hill Publication
2. **Hearn & Baker**, *Computer Graphics*, Prentice Hall India, Ltd.
3. **E. Balaguruswamy**, *Programming in ANSI C*, Tata McGraw Hill publication
4. **Y. Langsam, M.J. Augenstein, A.M. Tenenbaum,** *Data Structures Using C and C++*, Second Edition, Prentice Hall of India, 2000

## Practical Assignments

(Questions may not be restricted to this list)

1. Write a program to display the message “Welcome to the C programming world” on the screen.
2. Write a program to find out the sum of two integer values and display the result on the screen. Input the two values from the keyboard.
3. Write a program to find out the greatest of three numbers.
4. Write a program for **swap**ping the two numbers with / without using another variable.
5. Write a program to find whether the given year is a leap year or not ( use % modulus operator)
6. Write a program to find out the real roots of quadratic equation, $Ax^2+Bx+C=0$.
7. Write a program to convert the given temperature in Fahrenheit to Celsius using the following conversion formula, C=(F-32)/1.8.
8. Write a program to find out the average of any ten numbers. (Use (a) **while** loop, and (b) **for** loop).

9. Write a program to generate Fibonacci sequence. (1,1,2,3,5,8,13, …)
10. An employee is paid 1.5 times the normal rate for every hour beyond 40 hours worked in a week. Write a program to calculate the weekly wage of an employee.
11. Write a program to check whether the given string is palindrome or not.
12. The total distance traveled by a vehicle in t seconds is given by

    Distance = $ut + (at^2)/2$

    Where **u** is the initial velocity (meters per second), **a** is the acceleration (meters per second2). Write a program to evaluate the distance traveled at regular intervals of time, given the values of **u** and **a**. The program should provide the flexibility to the user to select his own time intervals and repeat the calculations for different values of **u** and **a**.
13. For a certain electrical circuit with an inductance L and resistance R, the damped natural frequency is given by

    Frequency $=\sqrt{[(1/LC-R^2/4C^2)]}$

    It is desired to study the variation of this frequency with C (capacitance). Write a program to calculate the frequency for different value of C starting from **0.1** in steps of **0.01**.
14. Write a program to read the following numbers, round them off to the nearest integers and print out the results in integer form:

    35.7 50.21 -23.73 -46.45
15. Given the string "WORDPROCESSING ", write a program to read the string from the terminal and display the same in the following formats:

    (a) WORD PROCESSING (b) WORD (c) W. P.
    PROCESSING
16. Admission to a professional course is subject to the following conditions:
    (a) Marks in mathematics >=60
    (b) Marks in physics >=50
    (c) Marks in chemistry >=40
    (d) Total in all three subjects >=200
    Or
    Total in mathematics and physics >=150
    Write a program to search of admission of students. The user has to enter the marks from the keyboard of the corresponding subjects.
17. Write a program that will read the value of **x** and evaluate the following function

    1 for x>0
    Y= 0 for x=0
    -1 for x<0

    Using
(a) nested **if** statements,
(b) **else if** statements, and
(c) Conditional operator?:

18. Write a program to calculate the monthly telephone bill according to the following rules:
    (a) Rural subscribers:

| | |
|---|---|
| Upto 250 calls | Free |
| 251 calls to 450 calls | 0.60 |
| 451 calls to 500 calls | 0.80 |
| 501 calls to 1000 calls | 1.00 |
| above 1000 calls | 1.20 |

(b) Urban subscribers:

| | |
|---|---|
| Upto 150 calls | Free |
| 151 calls to 400 calls | 0.80 |
| 401 calls to 1000 calls | 1.00 |
| Above 1000 calls | 1.20 |

(c) The rental for urban subscribers depends on the number of calls upto 400 calls the rental will be 200/- and above 400 calls the rental will be 240/-. For rural subscribers the rental is always 200/-.

19. Write a C program to input the Name, City Type (whether Metro or Non-Metro) and Basic Pay of an employee and calculate the salary according to the following rules:
    (a) Dearness allowance (DA)
    (i) Upto Rs. 3500 110% of basic pay
    (ii) Above Rs.3500 90% of the basic pay subject to a maximum of Rs. 3850 (i.e. DA should be at least Rs. 3850.
    (b) House Rent Allowance (HRA) is 15% of the basic pay subject to a maximum of Rs. 800 (i.e. never more than Rs. 800)
    (c) If City is Metro, City Compensatory Allowance (CCA)=800 else if it is Non-Metro, CCA=600.
    (d) Provident Fund (PF) is 12% of the basic pay.
    (**Total Salary**=Basic Pay +DA+HRA+CCA-PF)

    The **output** should be in the following format (Example only)

```
Example Name            ABCDEF
Basic Salary                  5000
Dearness Allowance            4500
HRA                     750
CCA: Non-Metro          600
PF                      600
Total Salary                  10250
```

20. Write a program to sum the following series:
    a) The first n natural numbers
    b) The first n odd natural numbers
    c) The first n even natural numbers
21. Write a program to sum the series : 2 * 3 – 3 * 5 + 4 * 7 + to n terms
22. Given a number, write a program using while loop to reverse the digits of the number. For example, the number 12345 should be written as 54321. (**Hint:** Use modulus operator to extract the last digit and the integer division by 10 to get the n-1 digit number from the n digit number.)
23. Write a program for sorting the elements of an array by using Selection sort, Bubble sort, Insertion sort.
24. Write a program to generate positive prime numbers.
25. Write a program to display the multiplication table of a given number from 1 to 20.
26. Write a program to display the multiplication table of a given number for a given range.
27. Write a program to display the multiplication table of a given group of numbers (maximum five numbers) for a given range.

21. Write a program to find the biggest and smallest number and its position in the given array.

22. Write a program to find addition, subtraction and multiplication of matrices using function.

23. The factorial of an integer m is the product of consecutive integers from 1 to m. That is,
Factorial m = m! =m*(m-1)*(m-2)*…*1.

24. Write a program to find the sum of row, column, and diagonals of the given matrix.
25. Write a program to find the largest number of the given matrix using function.
26. Write a program to sort all the elements of a matrix using function.
27. Write a program to input a string and perform the following tasks without using library functions: (a) to find its length, (b) to change it to upper case / lower case (c) to extract the left most n characters, (d) to extract the right most n characters (e) to extract n characters from it starting from position p, (f) to insert another string in it at position p (g) to replace n characters in it starting at position p with a given string
28. Write a program to search a pattern in a given text.
29. Write a program to search a pattern in a given text and replace every occurrence of it with another given string.
30. Write a program to write a given number in words using function.
31. Write a program to display the text in a FILE. (TYPE command in DOS).
32. Write a program to copy the contents of one text to another text file using command line arguments.
33. Write a program to merge the two text file to another text file.
34. Write a program to copy the contents of one text file to any number of given files using command line arguments.
35. Write a program to count the number of characters, lines and words in a text file.
36. Write a program to print every line of a text file containing a given pattern.
37. To copy a file by converting lower case text file to upper case text file using command line argument.
38. Write a program to input, sort, and display n names using using array of pointers.
39. Write a program to count the number of vowels, consonants, and other characters and the number of words in a string / file. A space, tab, or a punctuation mark separates a word (, ; . : !).
40. Write a menu driven program to create records of students with marks in various subjects and store them in a file (sequential, random or binary). Make provision for viewing all the records, searching a particular record, editing a particular record, deleting a particular record and listing a particular group of records.
41. Write a menu-driven program to
    a) Construct a singly linked list. Assume the information part of each node consists of only an integer key. Get input for each key from the keyboard. Assume the input is over when the user enters –1
    b) Print the information from each node
    c) Delete all nodes containing a given number
    d) Exit
42. Consider that *L*, a linked list of *n* integers, is given to you. Suppose, the nodes of the list are numbered from *1* to *n*. It is required to split the list *L* into 4 lists so that the first list contains the nodes of *L* numbered 1, 5, 9, 13 … The second list contains the nodes numbered 2, 6, 10, 14 … The third list contains the nodes of *L* numbered 3, 7, 11, 15, …. The fourth list contains the nodes of *L* numbered 4, 8, 12, 16 … Write a program to create the list and perform the splitting.
43. Write a C function to insert a node appropriately to an already sorted list so that after insertion, the new list also becomes sorted. Take care of special cases such as inserting into an empty list. Use this function to write a program which accepts integers at the input and at the end produces a sorted list. Assume that if the integer read at the input is ‘0’ then your program should stop.

44. Write a program to implement polynomial multiplication. Test your program by inputting the following two polynomials given below.
45. 10 P^8 + 14 P^6 - 8 P^5 - 3 P^4 + P^2
46. 3 P^ 4 + 5 P^3 - 2 P + 9
47. ( ^ is to be read as “raised to”)
48. Store each term of the polynomial in a linked list in descending order of the index. Use separate linked lists for each polynomial. Obtain and store the product in a third linked list, and then print out all the three polynomials in a format similar to the one shown above, in descending order of index.
49. A bi-directional list is a list of elements that are linked in both ways. Both links are originating from a header. Construct a module with procedures for searching, inserting and deleting elements.
50. Write a program to represent a sparse matrix using linked list. Add together two such matrices, and display the original and resulting matrices in matrix form.
51. Write a menu-driven program to implement a stack ***using arrays***. The menu should have the following options:
    a. Push on to the stack
    b. Pop from the stack and print the value popped from the stack
    c. Merely print the value on top of the stack
    d. Exit
52. Error trapping should be done for underflow and overflow. Available array space should be efficiently used (i.e. there cannot be overflow if there is more than 1 empty element in the array). Assume that the information part of a stack element is only an integer.
53. Write INSERT and DELETE functions in C language simulating insertion and deletion in circular queue which stores an array of characters.
54. A double-ended queue is a linear list in which additions and deletions may be made at either end of the queue. Write a C function to implement deque with desired functionality. Illustrate use of your function in an example problem, say, a queue of integers.
55. Devise a scheme to traverse a singly linked list in both directions by reversing the links during left to right traversal. Write a C program to implement this traversal scheme.
56. Write a C program to convert an expression from its infix form to its equivalent (a)postfix form, (b)prefix form. Assume the infix expression contains only operators +, -, /, *, ^. The operator ‘^’ stands for exponentiation. The operands are all single digit integers. Display the resulting (a)postfix expression (b)prefix expression.
57. Write a program to input a postfix expression that consists of only single digit positive operands and the binary operators **+, -, *,** and **/.** Using a function, evaluate this postfix expression. The function should report if the postfix expression is invalid, else return its value. [For example**, 242/-46*+7+** is a valid postfix expression (being the equivalent of the infix expression, **2-4/2+4*6+7**) and its value is 31.00.]
58. Write a program to construct a binary search tree of integers using linked list. Assume the information part of each node consists of only an integer key. Get input for each key from the keyboard. Assume the input is over when the user enters –1. Next, print out the keys in ascending order of magnitude, using a non-recursive function.
59. Write a program to create a binary tree and to traverse the tree in
    a. pre-order

b. in-order
c. post-order

60. Implement a procedure for deleting an element *X* from a binary search tree.
61. Write a program to reconstruct a binary search tree given its pre-order and in-order traversal sequence.
62. Write a program to find the biggest and smallest item in a binary search tree.
63. Design and implement an algorithm for insertion of an element in AVL tree taking into account all possible conditions.
64. Represent a graph using adjacency matrix. Write a C procedure to transform an adjacency matrix based representation to a linked-list based representation.
65. Design a suitable representation so that a graph can be stored on a hard disk. Write a procedure adjacency matrix based representation.
66. Write a program to represent a graph and perform a non-recursive depth first search of an item in it.
67. Write a program to represent a graph and compute the shortest distance between two nodes in it.
68. Write a program to input some numbers into an array, and then sort them using various sorting techniques (selection sort, bubble sort, quick sort, radix sort) and compare their time-complexities.
69. Write a program to input some numbers (at least 128 numbers, the more the better) from a file into two arrays A and B. Sort array B. Perform the linear search in array A and binary search in array B for a given number. Repeat these as many times as user decides and compare the time-complexity of the two search methods on the average.
70. Design and implement an algorithm to delete an identifier *X* from a hash table which uses hash function *f* and linear open addressing to resolve collisions. Your deletion scheme must ensure that correct search is possible even after deletion.