

ASSIGNMENT – 3

Programming for Problem Solving

Paper Code: ICT-101T

Unit III – Pointers, Structures, Unions, Dynamic Memory and File Handling | Maximum Marks: 50

Instructions: Attempt all questions. Draw neat flowcharts using standard symbols wherever asked. Write algorithms/pseudocode before implementation where specified. C programs should be properly indented, commented and logically tested.

Q1. Theory / Conceptual Understanding

- a) Explain pointer declaration, referencing and dereferencing with a suitable example. (2)
- b) Differentiate between a pointer to a pointer, an array of pointers and a generic (void) pointer. (3)
- c) Explain the need for self-referential structures. (2)

Bloom's Level: L2 – Understand

Q2. Dynamic Memory – Coding

Write a C program using dynamic memory allocation to accept N integers, find the largest and smallest values, and release the allocated memory correctly. Explain the role of malloc/calloc, realloc and free, and identify one possible memory-leak situation. (6)

Bloom's Level: L3 – Apply / L4 – Analyze

Q3. Pointers – Competency Based

Write C functions using pointers to swap two numbers, modify a variable through its address, and return a pointer from a function. Explain the lifetime issue that must be considered when a function returns a pointer. (5)

Bloom's Level: L3 – Apply / L4 – Analyze

Q4. Case Study – Employee Database

An organization maintains employee records containing ID, name, department and salary.

- a) Design a structure for the record. (1)
- b) Explain how an array of structures and a pointer to structure can be used. (2)
- c) Draw a flowchart for searching an employee by ID. (2)
- d) Write a C program to store N employees and display the employee with the highest salary. (4)

Bloom's Level: L3 – Apply / L4 – Analyze

Q5. Structures, Unions and Bit Fields – HOTS

An embedded system stores status information in a compact record. Analyze whether a structure, union or bit fields would be most appropriate for different parts of the record. Explain the effect of memory layout and alignment on the size of the record. Illustrate with a small C declaration using typedef and enum. (6)

Bloom's Level: L4 – Analyze / L5 – Evaluate

Q6. Polynomial Representation

Represent two polynomials using suitable C data structures and write functions for polynomial addition and multiplication. Explain how floating-point coefficients can be handled and how the representation affects the operations. (6)

Bloom's Level: L3 – Apply / L4 – Analyze

Q7. File Handling – Case-Based Coding

A student-result file contains student ID, name and marks. Design a C program to open the file, read records, calculate a result and write the processed information to another file. Include appropriate file modes and error checking using errno. State how the approach differs for text and binary files. (6)

Bloom's Level: L3 – Apply / L4 – Analyze

Q8. Advanced Pointer / Command-Line Problem

Design a command-line C program that accepts a filename as an argument and reports whether the file can be opened. Use argc/argv and appropriate error handling. Extend the design conceptually using a function pointer for selecting an operation on the file data. (5)

Bloom's Level: L4 – Analyze / L6 – Create

ASSIGNMENT – 2

Programming for Problem Solving

Paper Code: ICT-101T

Unit II – Arrays, Functions, Searching, Sorting and Strings | Maximum Marks: 50

Instructions: Attempt all questions. Draw neat flowcharts using standard symbols wherever asked. Write algorithms/pseudocode before implementation where specified. C programs should be properly indented, commented and logically tested.

Q1. Theory / Conceptual Understanding

- a) Differentiate between one-dimensional, two-dimensional and multi-dimensional arrays with suitable examples. (2)
- b) Explain call by value and the idea of call by reference in C. Why are arrays naturally suitable for passing to functions? (3)
- c) Explain the purpose of storage classes and distinguish between scope and lifetime. (2)

Bloom's Level: L2 – Understand

Q2. Arrays – Competency-Based

A class has N students and their marks are stored in an array. Write a C program using functions to calculate the highest mark, lowest mark, average mark and number of students scoring above the average. Explain how the array is passed to the functions. (6)

Bloom's Level: L3 – Apply / L4 – Analyze

Q3. Searching and Sorting – HOTS

A database contains 5,000 sorted roll numbers. Compare linear search and binary search for this application. Implement binary search in C and explain why the same algorithm is not directly suitable for an unsorted array. Also state one situation in which linear search may still be preferable. (6)

Bloom's Level: L4 – Analyze / L5 – Evaluate

Q4. Case Study – Student Record Analysis

A college needs to analyze marks of students in five subjects. Store the marks in a two-dimensional array where rows represent students and columns represent subjects.

- a) Design the data representation. (1)
- b) Develop an algorithm/pseudocode to calculate each student's total and percentage. (2)
- c) Draw a flowchart for processing one student's record and extending it to all students. (2)
- d) Write a C program using suitable loops. (4)

Bloom's Level: L3 – Apply / L4 – Analyze

Q5. Strings – Coding and Reasoning

Write a C program to read a string and determine whether it is a palindrome without using a direct palindrome-testing function. Then explain how the solution can be modified to ignore spaces and differences between uppercase and lowercase characters. (5)

Bloom's Level: L3 – Apply / L4 – Analyze

Q6. Recursion – Higher-Order Thinking

Write a recursive function to generate Fibonacci terms. Trace the function for N = 5. Compare the recursive and iterative approaches and identify why the recursive approach performs repeated computations. (5)

Bloom's Level: L4 – Analyze / L5 – Evaluate

Q7. Root Finding – Logical Problem Solving

Design an algorithm and draw a flowchart to find the square root of a positive floating-point number using an iterative approximation method. Clearly show initialization, repeated improvement and the stopping condition. Write a C implementation. (6)

Bloom's Level: L4 – Analyze / L5 – Evaluate

Q8. Sorting – Application

Write C functions for bubble sort, insertion sort and selection sort. For a small set of student marks, compare the number of major passes/iterations made by the three methods and recommend one method for the given situation with justification. (5)

Bloom's Level: L3 – Apply / L5 – Evaluate

ASSIGNMENT – 4

Programming for Problem Solving

Paper Code: ICT-101T

Unit IV – C Standard Libraries, Algorithms and Data Structures | Maximum Marks: 50

Instructions: Attempt all questions. Draw neat flowcharts using standard symbols wherever asked. Write algorithms/pseudocode before implementation where specified. C programs should be properly indented, commented and logically tested.

Q1. Theory / Conceptual Understanding

- a) Explain the concept of an Abstract Data Type (ADT) and distinguish an ADT from its implementation. (2)
- b) State the purpose of any four relevant C standard libraries from the syllabus and give one representative function/use for each. (3)
- c) Explain the purpose of assertions/diagnostics in program development. (2)

Bloom's Level: L2 – Understand

Q2. ADT and Linked List – Competency Based

Design a singly linked-list ADT supporting creation, traversal, insertion at the beginning and deletion from the beginning. Draw a flowchart for insertion at the beginning and write the required C functions. (6)

Bloom's Level: L3 – Apply / L4 – Analyze

Q3. Linked Lists – HOTS

A student-record system performs frequent insertions and deletions. Compare singly, doubly and circular linked lists. Recommend the most appropriate representation for (i) forward-only traversal, (ii) bidirectional traversal, and (iii) repeated cyclic traversal. Justify each choice and describe the required pointer changes. (6)

Bloom's Level: L4 – Analyze / L5 – Evaluate

Q4. Case Study – Browser Back and Forward

A web browser needs Back and Forward operations.

- a) Identify the suitable ADT/data-structure design. (1)
- b) Draw a flowchart showing the logic when a new page is visited and when Back is pressed. (3)
- c) Explain the role of push and pop operations. (2)
- d) Write suitable C functions to implement the stack operations. (3)

Bloom's Level: L4 – Analyze / L6 – Create

Q5. Stack – Expression Processing

For the infix expression $A + B * (C - D)$, convert it into postfix notation and show the important stack operations. Draw a flowchart for the core infix-to-postfix algorithm and write suitable C functions. Also explain how the same stack idea can be used for parenthesis balancing. (6)

Bloom's Level: L3 – Apply / L4 – Analyze

Q6. Queue – Application and Design

A hospital serves normal patients in arrival order but emergency patients must receive higher priority. Identify the suitable queue model and compare linear, circular and priority queues. Design enqueue/dequeue logic and write suitable C functions for the selected model. (6)

Bloom's Level: L4 – Analyze / L5 – Evaluate / L6 – Create

Q7. Trees and BST – Higher-Order Problem Solving

Construct a Binary Search Tree for: 50, 30, 70, 20, 40, 60, 80. Give inorder, preorder and postorder traversals. Explain searching and deletion of node 30, and write the C functions for BST insertion and search. (6)

Bloom's Level: L4 – Analyze / L5 – Evaluate

Q8. Modern C – Critical Thinking

Explain how `_Static_assert`, `_Generic`, function pointers, safer macros, forward declarations, program segmentation and header hygiene can improve reliability and maintainability of a C program. Propose a modular organization for a small data-structure program using separate header and source files. (5)

Bloom's Level: L4 – Analyze / L5 – Evaluate / L6 – Create

ASSIGNMENT – 1

Programming for Problem Solving

Paper Code: ICT-101T

Unit I – Introduction to Programming and Fundamentals of C | Maximum Marks: 50

Instructions: Attempt all questions. Draw neat flowcharts using standard symbols wherever asked. Write algorithms/pseudocode before implementation where specified. C programs should be properly indented, commented and logically tested.

Q1. Theory / Conceptual Understanding

- a) Explain the role of the preprocessor, compiler and linker in creating an executable C program. (3)
- b) Differentiate between an algorithm, flowchart and pseudocode with respect to their purpose in problem solving. (2)
- c) Explain the purpose of C tokens, keywords, identifiers and constants with suitable examples. (2)

Bloom's Level: L2 – Understand

Q2. Competency-Based – C Program Structure

A C program is required to accept two integers and display their sum, difference, product and quotient. Write the complete C program and identify the purpose of the header file, main() function, declarations, input/output statements and return statement. (5)

Bloom's Level: L3 – Apply

Q3. Logical Problem Solving through Flowchart

A positive integer is entered by the user. Develop an algorithm and draw a flowchart to determine whether the number is an Armstrong number. The solution should work for a number containing any number of digits. Write the corresponding C program. (6)

Bloom's Level: L3 – Apply / L4 – Analyze

Q4. Operators and Expressions – HOTS

Consider the expression: $a = b + c * d - e / f$. Explain the order in which the expression is evaluated using precedence and associativity rules. Then modify the expression using parentheses to make the intended order of evaluation explicit. Discuss the effect of type conversion if some operands are integers and others are floating-point values. (5)

Bloom's Level: L4 – Analyze

Q5. Case Study – Electricity Bill

A power utility wants a program to calculate an electricity bill based on units consumed. Different slabs have different per-unit rates. The program must accept units, calculate the bill using decision statements, and display the final amount.

- a) Identify inputs, processing and outputs. (2)
- b) Develop pseudocode and draw a flowchart for the decision-making logic. (3)
- c) Write the C program. (3)

Bloom's Level: L3 – Apply / L4 – Analyze

Q6. Loop-Based Problem Solving

Write a C program to accept N and display all prime numbers between 2 and N. Use loops and suitable decision logic. Explain how the loop terminates and how the program avoids unnecessary checks. (5)

Bloom's Level: L3 – Apply / L4 – Analyze

Q7. Decision Making – Competency

Write a C program that accepts three sides of a triangle and first checks whether they can form a valid triangle. If valid, determine whether the triangle is equilateral, isosceles or scalene. Draw a flowchart for the decision logic. (5)

Bloom's Level: L3 – Apply / L4 – Analyze

Q8. Higher-Order Logical Problem Solving

Design an algorithm and flowchart to find the second-largest number among N integers without sorting the complete array. Explain the logic used to maintain the largest and second-largest values, and write the C program. (7)

Bloom's Level: L4 – Analyze / L5 – Evaluate