

Programming in C

CUCS1001

Module 1: Basic Syntax and Control Structures (50 Questions)

1. Write a program to print "Hello, World!".
2. Write a program to declare and initialize variables of all basic data types.
3. Implement a program to find the size of int, float, char, and double data types.
4. Write a program to take user input for name, age, and marks, and display them.
5. Develop a program to check if a given number is even or odd.
6. Write a program to demonstrate the use of constants.
7. Implement a program to calculate the sum and average of three numbers.
8. Write a program to perform addition, subtraction, multiplication, and division of two numbers.
9. Develop a program to find the largest of three numbers.
10. Write a program to calculate the area of a rectangle.
11. Implement a program to convert a given temperature from Celsius to Fahrenheit.
12. Write a program to swap two numbers using a temporary variable.
13. Swap two numbers without using a temporary variable.
14. Write a program to demonstrate the use of logical operators.
15. Implement a program to find the square of a number using an arithmetic operator.
16. Write a program to check if a number is positive, negative, or zero.
17. Develop a program to calculate the area and circumference of a circle.
18. Write a program to evaluate a quadratic equation.
19. Implement a program to demonstrate the use of the ternary operator.
20. Write a program to convert kilometers to miles.
21. Implement a program to calculate simple interest.
22. Write a program to calculate compound interest.
23. Develop a program to check whether a year is a leap year.
24. Write a program to find the ASCII value of a character.
25. Implement a program to calculate the square root of a number.
26. Write a program to demonstrate the use of bitwise operators.
27. Implement a program to toggle the nth bit of a number.
28. Write a program to set the nth bit of a number.
29. Develop a program to count the number of set bits in a binary number.
30. Write a program to demonstrate the use of the modulus operator.
31. Implement a program to calculate the sum of digits of a number.
32. Write a program to reverse a given number.
33. Develop a program to check if a number is a palindrome.
34. Write a program to find the GCD of two numbers.
35. Implement a program to find the LCM of two numbers.
36. Write a program to calculate the power of a number using a loop.
37. Implement a program to demonstrate the use of relational operators.
38. Write a program to check if a character is an uppercase letter.
39. Develop a program to find the factorial of a number using a loop.
40. Write a program to demonstrate the use of assignment operators.
41. Implement a program to demonstrate type casting in C.
42. Write a program to calculate the sum of the first n natural numbers.
43. Develop a program to print the multiplication table of a given number.
44. Write a program to check if a character is a vowel or consonant.

45. Implement a program to check if a character is a digit.
46. Write a program to find the minimum and maximum of three numbers.
47. Develop a program to calculate the average of n numbers entered by the user.
48. Write a program to print the first n Fibonacci numbers.
49. Implement a program to find the remainder without using the % operator.
50. Write a program to calculate the product of two floating-point numbers.

Module 2: Control Statements (50 Questions)

1. Write a program using `if-else` to find the largest of three numbers.
2. Implement a program using `if-else` to check if a number is divisible by both 3 and 5.
3. Write a program to determine the grade of a student based on marks using `if-else`.
4. Develop a program using `nested if` to check if a number is positive and even.
5. Write a program to demonstrate the use of `switch-case` for basic calculator operations.
6. Implement a program to print the day of the week using `switch-case`.
7. Write a program to check if a character is a vowel using `switch-case`.
8. Develop a program to print numbers from 1 to 100 using a `for` loop.
9. Write a program to calculate the sum of the first n even numbers using a `for` loop.
10. Implement a program to find the factorial of a number using a `for` loop.
11. Write a program to reverse a given number using a `while` loop.
12. Develop a program to check if a number is prime using a `while` loop.
13. Write a program to print the Fibonacci series up to n terms using a `while` loop.
14. Implement a program to find the sum of digits of a number using a `do-while` loop.
15. Write a program to print a pyramid pattern using nested loops.
16. Develop a program to print the Pascal triangle using nested loops.
17. Write a program to check if a number is an Armstrong number.
18. Implement a program to print all prime numbers between two intervals.
19. Write a program to find the HCF of two numbers using a loop.
20. Develop a program to demonstrate the use of the `break` statement.
21. Write a program to demonstrate the use of the `continue` statement.
22. Implement a program to demonstrate the use of the `goto` statement.
23. Write a program to calculate the power of a number using recursion.
24. Develop a program to count the number of vowels in a string.
25. Write a program to check if a string is a palindrome.
26. Implement a program to find the length of a string without using library functions.
27. Write a program to concatenate two strings without using `strcat()`.
28. Develop a program to copy one string to another without using `strcpy()`.
29. Write a program to sort an array in ascending order using a loop.
30. Implement a program to merge two arrays.
31. Write a program to find the second largest number in an array.
32. Develop a program to find the smallest number in an array.
33. Write a program to calculate the transpose of a matrix.
34. Implement a program to add two matrices.
35. Write a program to multiply two matrices.
36. Develop a program to check if a matrix is symmetric.
37. Write a program to find the trace of a matrix.
38. Implement a program to rotate a matrix 90 degrees clockwise.
39. Write a program to find the determinant of a 2x2 matrix.
40. Develop a program to solve a linear equation using Cramer's rule.
41. Write a program to check if a number is perfect.
42. Implement a program to find the sum of prime numbers up to n.
43. Write a program to print the prime factors of a number.
44. Develop a program to find the nth term of an arithmetic progression.

45. Write a program to find the nth term of a geometric progression.
46. Implement a program to check if a number is a happy number.
47. Write a program to check if a number is a Kaprekar number.
48. Develop a program to find the sum of all divisors of a number.
49. Write a program to find the number of trailing zeroes in a factorial.
50. Implement a program to calculate the sum of squares of the first n natural numbers.

Module 3: Functions and Recursion (50 Questions)

1. Write a function to find the maximum of three numbers.
2. Implement a function to calculate the factorial of a number using recursion.
3. Write a function to check whether a number is prime.
4. Develop a program to find the GCD of two numbers using a function.
5. Write a program to calculate the LCM of two numbers using a function.
6. Implement a function to reverse a string.
7. Write a function to check if a string is a palindrome.
8. Develop a program to calculate the power of a number using a function.
9. Write a function to count the number of vowels in a string.
10. Implement a recursive function to generate the Fibonacci series.
11. Write a function to check if a number is an Armstrong number.
12. Develop a function to find the sum of digits of a number.
13. Write a recursive function to calculate the sum of the first n natural numbers.
14. Implement a function to calculate the sum of an array.
15. Write a function to find the maximum element in an array.
16. Develop a program to implement a basic calculator using functions.
17. Write a function to check if two strings are anagrams.
18. Implement a function to calculate the length of a string without using library functions.
19. Write a function to remove all vowels from a string.
20. Develop a program to find the transpose of a matrix using functions.
21. Write a function to add two matrices.
22. Implement a function to multiply two matrices.
23. Write a function to sort an array in ascending order.
24. Develop a program to merge two sorted arrays using functions.
25. Write a function to calculate the determinant of a 2x2 matrix.
26. Implement a function to rotate a matrix by 90 degrees clockwise.
27. Write a recursive function to solve the Tower of Hanoi problem.
28. Develop a function to check if a number is a happy number.
29. Write a function to find the nth term of an arithmetic progression.
30. Implement a function to find the nth term of a geometric progression.
31. Write a program to find the sum of squares of the first n natural numbers using recursion.
32. Develop a function to check if a number is a perfect number.
33. Write a function to print all prime factors of a number.
34. Implement a function to calculate the sum of all divisors of a number.
35. Write a function to find the second largest number in an array.
36. Develop a recursive function to print a pyramid pattern.
37. Write a function to count the frequency of each character in a string.
38. Implement a program to demonstrate the use of function overloading.
39. Write a function to swap two numbers using call by reference.
40. Develop a program to implement pass-by-value and pass-by-reference concepts.
41. Write a function to find the roots of a quadratic equation.
42. Implement a function to calculate the binomial coefficient using recursion.
43. Write a program to calculate combinations (nCr) using a function.
44. Develop a function to calculate permutations (nPr).

45. Write a function to generate a random number within a given range.
46. Implement a recursive function to find the sum of elements in a linked list.
47. Write a function to check if a number is a palindrome using recursion.
48. Develop a program to implement a function that accepts a variable number of arguments.
49. Write a function to convert a binary number to decimal.
50. Implement a program to find the HCF and LCM of two numbers using recursive functions.

Module 4: Arrays and Strings (50 Questions)

Arrays (25 Questions)

1. Write a program to find the largest and smallest elements in an array.
2. Implement a program to calculate the sum and average of elements in an array.
3. Write a program to perform linear search in an array.
4. Develop a program to perform binary search on a sorted array.
5. Write a program to sort an array using the bubble sort technique.
6. Implement a program to sort an array using the selection sort algorithm.
7. Write a program to merge two sorted arrays into a single sorted array.
8. Develop a program to reverse an array in place.
9. Write a program to find the second largest and second smallest elements in an array.
10. Implement a program to find the frequency of each element in an array.
11. Write a program to count the number of even and odd elements in an array.
12. Develop a program to find the sum of diagonal elements in a square matrix.
13. Write a program to transpose a matrix.
14. Implement a program to multiply two matrices.
15. Write a program to check if two matrices are equal.
16. Develop a program to find the row-wise and column-wise sums of a matrix.
17. Write a program to check if a matrix is symmetric.
18. Implement a program to rotate a matrix 90 degrees clockwise.
19. Write a program to add and subtract two matrices.
20. Develop a program to check if an array is a palindrome.
21. Write a program to find the maximum difference between two elements in an array.
22. Implement a program to shift elements of an array left or right by a given number of positions.
23. Write a program to check if two arrays are equal.
24. Develop a program to find pairs of elements in an array whose sum equals a given number.
25. Write a program to calculate the determinant of a 2x2 matrix.

Strings (25 Questions)

1. Write a program to find the length of a string without using built-in functions.
2. Implement a program to reverse a string without using built-in functions.
3. Write a program to count the number of vowels and consonants in a string.
4. Develop a program to check if a string is a palindrome.
5. Write a program to concatenate two strings without using built-in functions.
6. Implement a program to compare two strings without using built-in functions.
7. Write a program to copy one string to another without using built-in functions.
8. Develop a program to find the frequency of each character in a string.
9. Write a program to convert a string to uppercase without using built-in functions.
10. Implement a program to convert a string to lowercase without using built-in functions.
11. Write a program to count the number of words in a string.
12. Develop a program to find the first occurrence of a character in a string.

13. Write a program to find the last occurrence of a character in a string.
14. Implement a program to find and replace a word in a string.
15. Write a program to sort the characters of a string alphabetically.
16. Develop a program to extract a substring from a given string.
17. Write a program to remove all spaces from a string.
18. Implement a program to check if two strings are anagrams.
19. Write a program to find the longest word in a string.
20. Develop a program to capitalize the first letter of each word in a string.
21. Write a program to convert a string containing a number into an integer.
22. Implement a program to check if a string contains only digits.
23. Write a program to remove duplicate characters from a string.
24. Develop a program to count the number of occurrences of a word in a string.
25. Write a program to check if a string starts and ends with the same character.

Module 5: Pointers and Dynamic Memory Allocation (50 Questions)

1. Write a program to demonstrate the usage of pointers.
2. Implement a program to print the address of a variable using a pointer.
3. Write a program to swap two numbers using pointers.
4. Develop a program to add two numbers using pointers.
5. Write a program to find the largest element in an array using pointers.
6. Implement a program to reverse an array using pointers.
7. Write a program to concatenate two strings using pointers.
8. Develop a program to copy one string to another using pointers.
9. Write a program to compare two strings using pointers.
10. Implement a program to find the length of a string using pointers.
11. Write a program to count vowels in a string using pointers.
12. Develop a program to demonstrate pointer arithmetic.
13. Write a program to create a dynamic array and input its elements.
14. Implement a program to find the sum of elements in a dynamic array.
15. Write a program to resize a dynamic array using `realloc`.
16. Develop a program to free memory allocated dynamically.
17. Write a program to create a 2D array dynamically.
18. Implement a program to find the sum of diagonal elements of a dynamically allocated matrix.
19. Write a program to transpose a matrix using pointers.
20. Develop a program to demonstrate the concept of a pointer to a pointer.
21. Write a program to sort an array dynamically using pointers.
22. Implement a program to find the largest and smallest elements in a dynamic array.
23. Write a program to reverse a string dynamically using pointers.
24. Develop a program to swap two strings using pointers.
25. Write a program to demonstrate the concept of function pointers.
26. Implement a program to pass a function as an argument to another function.
27. Write a program to implement a callback function using pointers.
28. Develop a program to demonstrate an array of function pointers.
29. Write a program to find the factorial of a number using a pointer to a function.
30. Implement a program to find the GCD of two numbers using a pointer to a function.
31. Write a program to demonstrate the use of void pointers.
32. Develop a program to demonstrate the use of NULL pointers.
33. Write a program to check if two strings are anagrams using pointers.
34. Implement a program to find the frequency of characters in a string using pointers.
35. Write a program to dynamically allocate memory for an array of strings.
36. Develop a program to concatenate two strings dynamically.
37. Write a program to create a structure dynamically using pointers.

38. Implement a program to access structure members using pointers.
39. Write a program to sort an array of structures using pointers.
40. Develop a program to swap two structures using pointers.
41. Write a program to implement a basic string library using pointers.
42. Implement a program to count the occurrence of a word in a dynamic string.
43. Write a program to tokenize a string dynamically using pointers.
44. Develop a program to implement a basic dynamic memory pool.
45. Write a program to demonstrate a memory leak and its prevention.
46. Implement a program to find the sum of two numbers using double pointers.
47. Write a program to demonstrate a wild pointer and its resolution.
48. Develop a program to use a pointer to access an array of structures.
49. Write a program to implement dynamic memory allocation for a 3D array.
50. Implement a program to find the sum of elements in a jagged array dynamically.

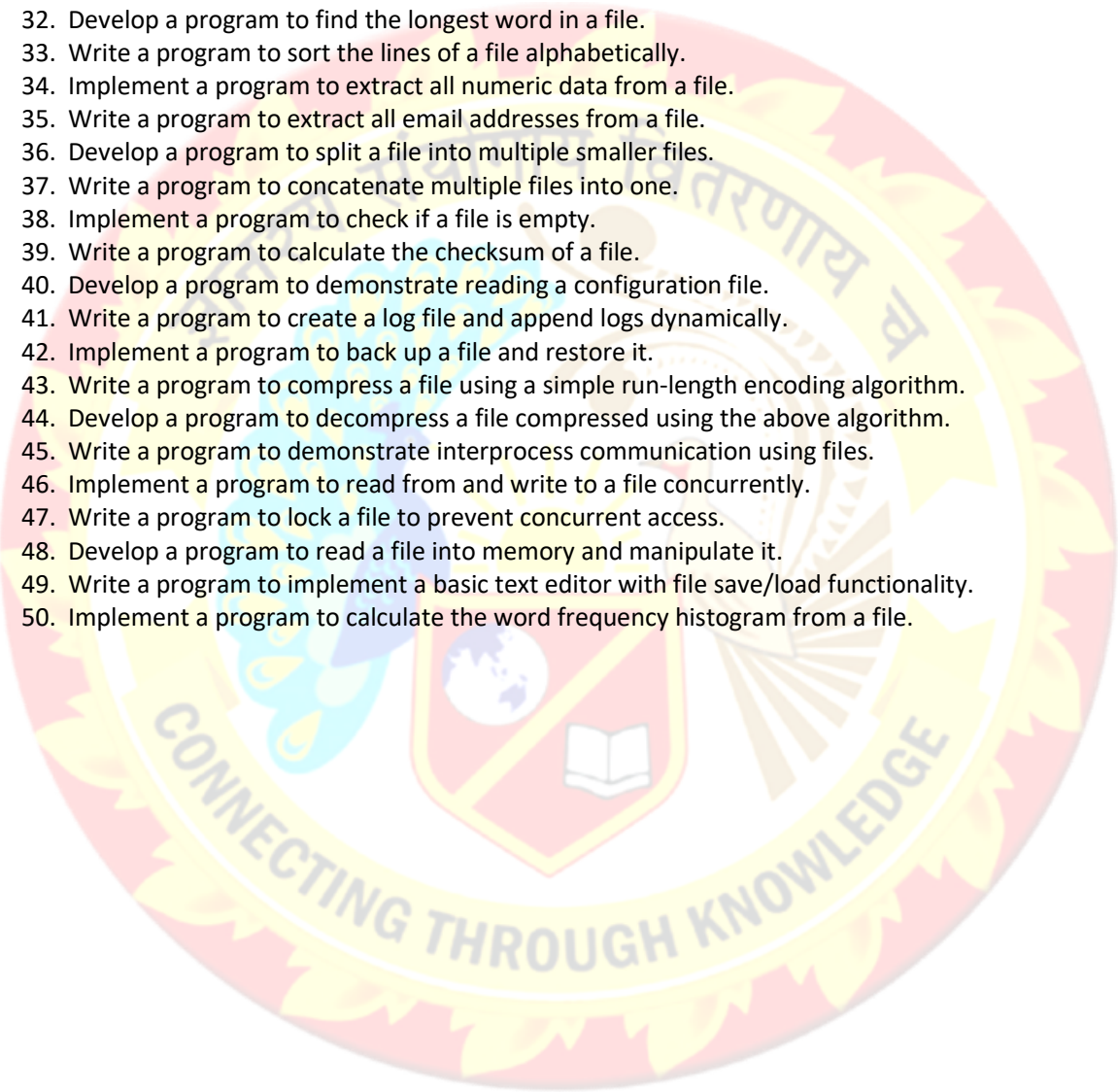
Module 6: Structures, Unions, and Enums (50 Questions)

1. Write a program to define and use a structure.
2. Implement a program to read and display student details using structures.
3. Write a program to calculate the total and average marks of students using structures.
4. Develop a program to store and display employee details using an array of structures.
5. Write a program to find the oldest and youngest employee using structures.
6. Implement a program to calculate and display the salary of employees using structures.
7. Write a program to sort employee records by their names using structures.
8. Develop a program to merge two arrays of structures.
9. Write a program to demonstrate nested structures.
10. Implement a program to calculate the distance between two points using structures.
11. Write a program to create and use an array of structures containing details of books.
12. Develop a program to find a book by its title in an array of structures.
13. Write a program to store and display the details of a cricket team using structures.
14. Implement a program to demonstrate the use of typedef with structures.
15. Write a program to use a structure to represent a date and calculate the difference between two dates.
16. Develop a program to create and use a structure for a complex number and perform addition and subtraction.
17. Write a program to implement a student grading system using structures.
18. Implement a program to swap two structures using a temporary structure.
19. Write a program to copy the contents of one structure into another structure.
20. Develop a program to demonstrate passing a structure to a function.
21. Write a program to return a structure from a function.
22. Implement a program to dynamically allocate memory for a structure.
23. Write a program to read and display details of products using structures.
24. Develop a program to find the product with the highest price using structures.
25. Write a program to demonstrate the use of unions.
26. Implement a program to store and display data of different types using a union.
27. Write a program to calculate and display size differences between a structure and a union.
28. Develop a program to demonstrate the use of a union to manage memory.
29. Write a program to use a union for storing data for multiple data types in a variable.
30. Implement a program to use both structures and unions in a single program.
31. Write a program to define and use an enum.
32. Develop a program to represent days of the week using an enum.
33. Write a program to represent months of the year using an enum and calculate the number of days in a given month.
34. Implement a program to use enums to represent error codes.

35. Write a program to calculate the total bill using enums to represent item categories.
36. Develop a program to demonstrate the use of enums in a switch-case statement.
37. Write a program to use enums and structures together.
38. Implement a program to represent a deck of cards using structures and enums.
39. Write a program to create a structure for a bank account and perform deposit and withdrawal operations.
40. Develop a program to use structures for managing a library system.
41. Write a program to define a structure for a movie ticket and calculate the total cost of tickets booked.
42. Implement a program to use structures for inventory management in a store.
43. Write a program to create a structure for a car and calculate its depreciation.
44. Develop a program to use a structure to store and sort employee details based on their age.
45. Write a program to manage student records using structures and perform CRUD (Create, Read, Update, Delete) operations.
46. Implement a program to calculate the area of different shapes using a structure with a union.
47. Write a program to calculate electricity bills using structures to store customer details.
48. Develop a program to demonstrate a menu-driven program using structures for an address book.
49. Write a program to compare two dates using structures.
50. Implement a program to represent a chessboard using structures and enums.

Module 7: Advanced Topics in C (50 Questions)

1. Write a program to open a file in read mode.
2. Implement a program to write data to a file.
3. Write a program to append data to an existing file.
4. Develop a program to read a file character by character.
5. Write a program to read a file line by line.
6. Implement a program to count the number of characters in a file.
7. Write a program to count the number of words in a file.
8. Develop a program to count the number of lines in a file.
9. Write a program to check if a file exists.
10. Implement a program to copy the contents of one file to another.
11. Write a program to reverse the contents of a file.
12. Develop a program to compare two files.
13. Write a program to merge two files into one.
14. Implement a program to delete a file.
15. Write a program to rename a file.
16. Develop a program to encrypt a file using a simple encryption algorithm.
17. Write a program to decrypt a file encrypted using the above program.
18. Implement a program to find the size of a file.
19. Write a program to create a file dynamically and write data to it.
20. Develop a program to read and write data using formatted I/O.
21. Write a program to read and write binary data to a file.
22. Implement a program to create a file and write an array of structures to it.
23. Write a program to read an array of structures from a file.
24. Develop a program to update a record in a file.
25. Write a program to delete a record from a file.
26. Implement a program to demonstrate random file access using `fseek`.
27. Write a program to demonstrate the use of `ftell` and `rewind`.
28. Develop a program to count the frequency of a word in a file.
29. Write a program to count the occurrence of a character in a file.
30. Implement a program to search for a specific word in a file.
31. Write a program to replace a word in a file with another word.

- 
- The logo of Centurion University is a circular emblem. It features a central shield with a globe and an open book. The shield is surrounded by a yellow border with the text "CONNECTING THROUGH KNOWLEDGE". The outermost ring of the logo contains the university's name in Hindi: "सेण्टुरियन विश्वविद्यालय".
32. Develop a program to find the longest word in a file.
 33. Write a program to sort the lines of a file alphabetically.
 34. Implement a program to extract all numeric data from a file.
 35. Write a program to extract all email addresses from a file.
 36. Develop a program to split a file into multiple smaller files.
 37. Write a program to concatenate multiple files into one.
 38. Implement a program to check if a file is empty.
 39. Write a program to calculate the checksum of a file.
 40. Develop a program to demonstrate reading a configuration file.
 41. Write a program to create a log file and append logs dynamically.
 42. Implement a program to back up a file and restore it.
 43. Write a program to compress a file using a simple run-length encoding algorithm.
 44. Develop a program to decompress a file compressed using the above algorithm.
 45. Write a program to demonstrate interprocess communication using files.
 46. Implement a program to read from and write to a file concurrently.
 47. Write a program to lock a file to prevent concurrent access.
 48. Develop a program to read a file into memory and manipulate it.
 49. Write a program to implement a basic text editor with file save/load functionality.
 50. Implement a program to calculate the word frequency histogram from a file.

Centurion

UNIVERSITY

Shaping Lives...

Empowering Communities...