Government (Auto.) college, Rourkela

SUBJECT-STATISTICS

PAPER- CORE-12

SHORT TYPE QUESTIONS

1. List three basic data types in C.
2. Differentiate between int and float data types in C
3. Define a constant and provide an example.
4. List three arithmetic operators in C.
5. Differentiate between a relational operator and a logical operator.
6. How do you increment a variable's value in C? Provide an example.
7. Explain operator precedence and give an example of its importance in an expression.
8. What is implicit type conversion in C? Give an example.
9. How can you explicitly convert a variable from one data type to another?
10. Provide two examples of library functions in C and briefly explain their usage.
11. Explain the purpose of printf() and scanf() functions in C.
12. How is data read from the user in C? Name a function used for this purpose.
13. How does the scanf() function work? Provide an example.
14. The ________________ library function can be used to convert a string to a floating-point number.
15. _______________ are names given to various program elements such as variables, functions, arrays, etc., in C programming.
16. An identifier must start with a ________________ or ________________ and can be followed by letters, digits, or underscores.
17. The process of running a C program is called program ________________.
18. What is the purpose of decision-making structures in C programming?
19. Differentiate between the "if" statement and the "if...else" statement.
20. Explain the concept of nesting in the context of "if...else" statements.
21. What is the purpose of an "else if" ladder? Provide an example scenario where it can be useful.
22. How does the "switch" statement differ from a series of "if" statements?
23. Briefly describe the conditional operator (ternary operator) and its usage in C.
24. What is the purpose of looping structures in programming?
25. How do you declare and initialize a one-dimensional array in C?
26. Explain the concept of a two-dimensional array. Provide an example use case.
27. What is a character array, and how is it different from a string?
28. Describe the process of declaring and initializing a string variable in C.
29. How can you read a string from the terminal using the "scanf" function?
30. The ________________ statement provides an alternative to using a series of "if" statements for decision making.
31. The ________________ operator is a shorthand way of writing an "if...else" statement in a single line.
32. A ________________ loop repeats a block of code as long as a certain condition is true.
33. In a "for" loop, the three components are initialization, condition, and ________________.

34. The "do...while" loop guarantees that the loop body is executed ________________ before checking the condition.
35. To prematurely exit a loop, the ________________ statement can be used.
36. A two-dimensional array is an array of arrays, where each element can be accessed using ________________ indices.
38. What is the purpose of using user-defined functions in a C program?
39. How do you define a function in C? Mention the components of a function definition.
40. What is the significance of a return value in a function? Which keyword is used to return a value from a function?
41. Give an example of a function with no arguments and no return values.
42. Explain the concept of a function with arguments but no return value. Provide an example.
43. How do you define a function with arguments and a return value? Mention the steps to call this type of function.
44. What is the purpose of a function that has no arguments but returns a value?
45. How do you declare and use a function that returns multiple values?
46. To call a function in C, you use its name followed by parentheses containing the ________________ (if required) that are passed to the function.
47. A function that has arguments but no return value is useful for performing operations that ________________ data but do not produce a result.
48. Functions with arguments and return values are used when you need to pass data to the function and receive a ________________ based on the computation.
49. A function with no arguments but returning a value can be employed to calculate and provide a result based on ________________ data.
50. Functions that return ________________ values enable you to send multiple pieces of information back to the caller using pointers or structures.
51. What is R used for in programming and data analysis?
52. How can you acquire and install R on your system?
53. What is the purpose of command packages in R?
54. How do you read data into R from an external source?
55. Name two types of data items in R.
56. What is the purpose of examining the structure of data items in R?
57. How do you save your work in R for future use?
58. The function ________________() is used to view the contents of an object.
59. In R, a vector is an example of a ________________ data item.
60. The ________________ function is used to check the structure of a data frame.
61. To create a new data frame in R, you can use the function ________________().

Long-Type Questions:

1. Discuss the historical background of the C programming language and its significance in the world of programming.
2. Explain the components of a C program and their roles in creating a structured program.
3. Describe the basic structure of a C program, including the main() function and its purpose.
4. Differentiate between keywords and identifiers in C programming. Provide examples of each.
5. Define basic data types in C and provide examples of their usage.
6. Enumerated data types allow programmers to create user-defined types. Explain their importance and provide an example.
7. What are derived data types in C? Discuss the role of arrays and structures in creating derived data types.
8. Explain the concepts of constants and variables in C programming. Provide examples of constant and variable declarations.
9. Discuss the various categories of operators in C (arithmetic, relational, logical, etc.) and provide examples of each.
10. Explain the concept of operator precedence in C. Provide an example where operator precedence affects the outcome of an expression.
11. What is type conversion in C? Differentiate between implicit and explicit type conversions, providing examples of each.
12. How can library functions enhance the functionality of a C program? Provide examples of commonly used library functions.
13. Create a C program that takes two integers as input and swaps their values using a temporary variable.
14. Develop a C program to check if a given number is even or odd and display an appropriate message.
15. Develop a C program that calculates the sum of all even numbers from 1 to N, where N is taken as input.
16. Develop a program that reads a string from the user and counts the number of vowels and consonants.
17. Write a C program that takes a positive integer as input and prints its reverse.
18. Write a C program to determine if a given number is positive, negative, or zero using if...else statements.
19. Create a program that takes a character as input and checks whether it's a vowel or a consonant using a switch statement.
20. Write a C program to find the maximum of three numbers using nested if...else statements.
21. Create a C program to calculate the factorial of a number using a while loop.
22. Write a program that prints the Fibonacci series up to a user-defined limit using a do...while loop.
23. Implement a C program that reads elements into a one-dimensional array and calculates their sum and average.
24. Write a C program to reverse a string using character arrays and loops.

25. Develop a program that takes a sentence as input and counts the number of words in it.
26. Explain the concept of decision making in programming. Discuss the roles of "if...else" statements, "else if" ladder, and the "switch" statement in making decisions in C.
27. Describe the different forms of loops available in C programming: "for," "while," and "do...while." Provide examples of scenarios where each type of loop might be used effectively.
28. Discuss the significance of arrays in programming. Explain the process of declaring and initializing a one-dimensional array in C. Provide an example of its usage.
29. How can nested "for" loops be used to generate patterns in programming? Provide a detailed example of a pattern that can be generated using nested loops.
30. What is the purpose of character arrays in C? Explain the process of declaring and initializing a character array. How can character arrays be used to work with strings?
31. Develop a program that uses arrays to store the temperatures of each day in a week. Calculate and display the average temperature for the week using a loop.
32. Create a C program that converts a given sentence to uppercase letters using character arrays and a loop. Display the modified sentence.
33. Explain the concept of user-defined functions in C programming. Why are they important for creating modular and maintainable code?
34. Describe the process of defining a function in C, including the function signature, return type, parameters, and function body.
35. Discuss the role of function prototypes in C programming. Why are they necessary, and how do they help in resolving issues related to function calls?
36. Differentiate between functions that have no arguments and no return values and functions that have arguments but no return values. Provide examples of each.
37. How can you pass arguments to a function in C? Explain the concept of pass by value and pass by reference, providing suitable examples.
38. Elaborate on the use of functions with arguments and return values. Provide scenarios where such functions are advantageous.
39. Describe the steps involved in creating a function that returns multiple values. Provide an example of a function that returns both the sum and product of two numbers.
40. Discuss the role of recursion in C programming and provide an example of a recursive function.
41. Explain the importance of R as a programming language for data analysis. How does it differ from traditional spreadsheet software?
42. Explain the purpose of command packages in R. How do you install and load packages, and how can they enhance your data analysis capabilities?
43. Starting with reading and getting data into R, discuss the various methods to import data from different file formats, such as CSV and Excel.
44. Describe the concept of named objects in R. How can you create, assign values to, and manipulate these objects?
45. Differentiate between the types of data items in R, including atomic vectors, lists, matrices, and data frames. Provide examples of each.
46. Discuss the structure of data items in R, focusing on the hierarchical arrangement of data within different data structures.

47. Explain the process of examining the structure of data items using functions like "str()" and "summary()" in R.
48. Describe the methods to save your work in R for future use. How can you save and load data, scripts, and workspaces?
49. Discuss the various techniques for manipulating objects in R, such as indexing, subsetting, and reordering elements within data structures.
50. Describe the steps involved in constructing different types of data objects, including creating vectors, matrices, and data frames.
51. Discuss the concept of descriptive statistics in data analysis. How can measures like mean, median, and standard deviation help in summarizing data?
52. Write an R program that reads a CSV file containing sales data, calculates the total sales for each month, and displays the results.
53. Create an R script that loads the "iris" dataset, subsets it to include only the species "setosa," and calculates the mean petal length.
54. Develop an R program that generates a simple bar plot to visualize the frequency distribution of ages in a dataset.
55. Use R to create a scatter plot from a given dataset and add a trendline to visualize the relationship between two variables.
56. Write an R script that calculates the correlation coefficient between two numeric variables in a dataset.
57. Develop a program that reads a text file, counts the occurrences of each word, and displays a word frequency table.
58. Create an R function that calculates the factorial of a given positive integer using a loop.
59. Develop a program that reads a CSV file containing student grades, calculates the average grade for each student, and categorizes their performance.
60. Create a C program that prints a pattern of asterisks in the shape of a right-angled triangle using nested loops.