Multiple choice Questions and Answers

1. What is the primary difference between text and binary files in C?

   a. Text files store data as plain text, while binary files store data in a compressed format.

   b. Text files are only readable by humans, while binary files are machine-readable.

   c. Text files use a newline character to represent line breaks, while binary files store data in bytes without special characters.

   d. Text files support random access, while binary files only allow sequential access.

   Answer: c. Text files use a newline character to represent line breaks, while binary files store data in bytes without special characters.

2. How do you open a file named "example.txt" in C for writing text data?

   a. fopen("example.txt", "w")

   b. fopen("example.txt", "r")

   c. fopen("example.txt", "a")

   d. fopen("example.txt", "wb")

   Answer: a. fopen("example.txt", "w")

3. Which function is used to write a character to a text file in C?

   a. putc()

   b. fputc()

   c. write()

   d. fwrite()

   Answer: b. fputc()

4. In C, what function is used to read a line of text from a file?

   a. fgets()

b. gets()

c. readline()

d. getline()

Answer: a. fgets()

5. How do you open a file in binary mode for reading and writing simultaneously in C?

a. fopen("file.bin", "rw")

b. fopen("file.bin", "r+")

c. fopen("file.bin", "rb+")

d. fopen("file.bin", "w+b")

Answer: c. fopen("file.bin", "rb+")

6. What is the purpose of the fprintf function in C when working with files?

a. Reads data from a file.

b. Appends data to a file.

c. Writes formatted data to a file.

d. Closes a file.

Answer: c. Writes formatted data to a file.

7. Which function is used to append data to an existing file in C?

a. fopen("file.txt", "a")

b. fopen("file.txt", "r+")

c. fopen("file.txt", "ab")

d. fopen("file.txt", "a+")

Answer: a. fopen("file.txt", "a")

8. How do you move the file pointer to a specific position in a file in C?

a. fsetpos()

b. fmove()

c. fseek()

d. fmovepos()

Answer: c. fseek()

9. In C, which function is used to read a block of data from a file?

a. fread()

b. read()

c. fgetline()

d. fgets()

Answer: a. fread()

10. What is the purpose of the rewind function in C file handling?

a. Closes the file.

b. Moves the file pointer to the beginning of the file.

c. Deletes the file.

d. Appends data to the file.

Answer: b. Moves the file pointer to the beginning of the file.

11. How do you write a structure to a binary file in C?

a. fwrite(structVar, sizeof(structVar), 1, filePtr);

b. write(filePtr, structVar, sizeof(structVar));

c. fputs(structVar, filePtr);

d. fprintf(filePtr, structVar);

Answer: a. fwrite(structVar, sizeof(structVar), 1, filePtr);

12. What does the feof function in C file handling indicate?

a. Checks if the file is open.

b. Checks if the end-of-file indicator is set.

c. Checks if the file exists.

d. Checks if the file is empty.

Answer: b. Checks if the end-of-file indicator is set.

13. Which function is used to check for errors in file operations in C?

a. ferror()

b. error()

c. checkError()

d. fileError()

Answer: a. ferror()

14. How do you close a file in C using the fclose function?

a. closeFile(filePtr);

b. close(filePtr);

c. fclose(filePtr);

d. filePtr.close();

Answer: c. fclose(filePtr);

15. What is the purpose of the remove function in C file handling?

a. Deletes a file.

b. Removes a character from a file.

c. Removes a line from a file.

d. Removes a directory.

Answer: a. Deletes a file.

16. In C, what is the significance of the binary mode while working with files?

a. It compresses the file.

b. It enables random access.

c. It treats the file as plain text.

d. It preserves the exact byte representation of data.

Answer: d. It preserves the exact byte representation of data.

17. How do you read a structure from a binary file in C?

a. fread(structVar, sizeof(structVar), 1, filePtr);

b. read(filePtr, structVar, sizeof(structVar));

c. fgets(structVar, filePtr);

d. fscanf(filePtr, structVar);

Answer: a. fread(structVar, sizeof(structVar), 1, filePtr);

18. Which function is used to set the file position indicator to the beginning of a file in C?

a. rewind()

b. fsetpos()

c. fseek(filePtr, 0, SEEK_SET);

d. fmovepos(filePtr, 0);

Answer: a. rewind()

19. What does the ftell function in C file handling return?

a. Current file size.

b. Current file position.

c. Number of lines in the file.

d. Number of characters in the file.

Answer: b. Current file position.

20. How do you open a file in C to read and write binary data without truncating the file?

    a. fopen("file.bin", "w+b")

    b. fopen("file.bin", "r+b")

    c. fopen("file.bin", "wb+")

    d. fopen("file.bin", "ab+")

    Answer: b. fopen("file.bin", "r+b")

21. What is the purpose of the fgetpos function in C file handling?

    a. Reads a line from a file.

    b. Gets the current file position.

    c. Appends data to a file.

    d. Reads a character from a file.

    Answer: b. Gets the current file position.

22. How do you write a string to a text file in C using the fputs function?

    a. fputs(filePtr, "Hello, World!");

    b. fwrite("Hello, World!", 1, strlen("Hello, World!"), filePtr);

    c. fprintf(filePtr, "Hello, World!");

    d. fputs("Hello, World!", filePtr);

    Answer: d. fputs("Hello, World!", filePtr);

23. Which function is used to read a character from a file in C?

    a. fread()

    b. fgetc()

    c. getc()

    d. readChar()

Answer: b. fgetc()

24. How do you check if a file exists in C before opening it?

    a. file_exists("file.txt");

    b. exists(filePtr);

    c. fexist("file.txt");

    d. access("file.txt", F_OK);

    Answer: d. access("file.txt", F_OK);

25. In C, what is the purpose of the clearerr function in file handling?

    a. Clears the screen.

    b. Clears the file.

    c. Clears the error indicator for a file.

    d. Clears the buffer.

    Answer: c. Clears the error indicator for a file.

26. What is the purpose of declaring a function prototype in C?

    a. To define the function

    b. To declare the function

    c. To call the function

    d. To initialize the function

    Answer: b. To declare the function

27. What is the signature of a function?

    a. Return type, name, and parameters

    b. Function body

    c. Function name and return type

    d. Parameters and return type

Answer: a. Return type, name, and parameters

28. In C, how are function parameters passed by default?

    a. Call by value

    b. Call by reference

    c. Call by pointer

    d. Call by name

    Answer: a. Call by value

29. Which term is used for the variable names in a function definition?

    a. Local variables

    b. Global variables

    c. Formal parameters

    d. Actual parameters

    Answer: c. Formal parameters

30. What is the purpose of the return type in a function declaration?

    a. To indicate the function's name

    b. To specify the function's signature

    c. To indicate the data type of the return value

    d. To specify the function's visibility

    Answer: c. To indicate the data type of the return value

31. What does "void" as a return type in a function declaration indicate?

    a. The function returns no value

    b. The function returns an integer

    c. The function returns a character

    d. The function returns a pointer

Answer: a. The function returns no value

32. In C, how are arrays passed to functions?

    a. Call by value

    b. Call by reference

    c. Call by pointer

    d. Call by name

    Answer: b. Call by reference

33. What is the idea of "call by reference" in C functions?

    a. Function parameters are passed by value

    b. Function parameters are passed by reference

    c. Function parameters are passed by pointer

    d. Function parameters are passed by name

    Answer: b. Function parameters are passed by reference

34. Which standard library function is used to find the length of a string?

    a. strlen()

    b. strlength()

    c. stringlen()

    d. length()

    Answer: a. strlen()

35. What is the purpose of the "stdlib.h" library in C?

    a. Input/output operations

    b. Memory allocation and deallocation

    c. String manipulation

    d. Time and date functions

Answer: b. Memory allocation and deallocation

36. Which function is used to allocate memory dynamically in C?

    a. malloc()

    b. allocate()

    c. dynamic()

    d. memalloc()

    Answer: a. malloc()

37. How are pointers passed to functions in C?

    a. Call by value

    b. Call by reference

    c. Call by pointer

    d. Call by name

    Answer: a. Call by value

38. What is the purpose of the "math.h" library in C?

    a. Input/output operations

    b. String manipulation

    c. Mathematical functions

    d. File operations

    Answer: c. Mathematical functions

39. Which of the following is an example of a call by reference in C?

    a. Passing an array

    b. Passing a structure

    c. Passing a character

    d. Passing an integer

Answer: a. Passing an array

40. What does the "void" keyword indicate in a function parameter list?

    a. No parameters are passed

    b. The parameter is a void type

    c. The parameter is a pointer

    d. The parameter is a character type

    Answer: a. No parameters are passed

    Explanation: In a function parameter list, "void" indicates that the function takes no parameters.

41. In C, which library is used for handling character input/output?

    a. stdlib.h

    b. math.h

    c. stdio.h

    d. string.h

    Answer: c. stdio.h

42. What is the purpose of the "ctype.h" library in C?

    a. String manipulation

    b. Time and date functions

    c. Character classification and conversion

    d. File operations

    Answer: c. Character classification and conversion

43. Which function is used to convert a string to an integer in C?

    a. atoi()

    b. itoa()

c. stoi()

d. str2int()

Answer: a. atoi()

44. How are structures passed to functions in C?

a. Call by value

b. Call by reference

c. Call by pointer

d. Call by structure

Answer: a. Call by value

45. What is the purpose of the "assert.h" library in C?

a. String manipulation

b. Time and date functions

c. Program assertion

d. File operations

Answer: c. Program assertion

46. Which function is used to perform dynamic memory allocation in C?

a. malloc()

b. allocate()

c. memalloc()

d. dynamic()

Answer: a. malloc()

47. What is the significance of the "const" keyword in function parameters?

a. The parameter is a constant

b. The parameter is a variable

c. The parameter is a pointer

d. The parameter is a structure

Answer: a. The parameter is a constant

48. What is the purpose of the "time.h" library in C?

a. Input/output operations

b. String manipulation

c. Time and date functions

d. Mathematical functions

Answer: c. Time and date functions

49. How are multidimensional arrays passed to functions in C?

a. Call by value

b. Call by reference

c. Call by pointer

d. Call by name

Answer: b. Call by reference

50. Which function is used to find the square root of a number in C?

a. sqrt()

b. pow()

c. sqr()

d. root()

Answer: a. sqrt()

51. What is recursion in programming?

a. A loop construct

b. A programming language

c. A function calling itself

d. A data type

Answer: c. A function calling itself

52. Which of the following is an example of a recursive problem?

a. Sorting an array

b. Finding the maximum value in an array

c. Calculating the factorial of a number

d. Adding two numbers

Answer: c. Calculating the factorial of a number

53. What is the base case in a recursive function?

a. The starting point of recursion

b. The condition that stops the recursion

c. The recursive call

d. The final result of the recursion

Answer: b. The condition that stops the recursion

54. Which of the following is a limitation of recursive functions?

a. Easy to understand

b. Efficient memory usage

c. Limited stack space

d. Suitable for all types of problems

Answer: c. Limited stack space

55. What is the Fibonacci series?

a. A series of random numbers

b. A series of prime numbers

c. A series where each number is the sum of the two preceding ones

d. A geometric series

Answer: c. A series where each number is the sum of the two preceding ones

56. What is the recursive formula for calculating the factorial of a number (n)?

a. n * (n-1)!

b. n!

c. n + factorial(n-1)

d. factorial(n-1) / n

Answer: a. n * (n-1)!

57. What is the significance of the base case in recursive factorial calculation?

a. It defines the starting point of the calculation.

b. It prevents an infinite loop.

c. It is where the final result is obtained.

d. It determines the recursive formula.

Answer: b. It prevents an infinite loop.

58. What is tail recursion?

a. A type of recursion that involves animal tails

b. A recursive function where the recursive call is the last operation

c. A recursion with no base case

d. A recursion with a non-recursive function call

Answer: b. A recursive function where the recursive call is the last operation

59. In the context of recursion, what is a stack overflow?

a. A condition where the base case is not reached

b. A condition where the recursion is too deep, exhausting the stack space

c. A condition where the recursion is too shallow

d. A condition where the base case is reached too quickly

Answer: b. A condition where the recursion is too deep, exhausting the stack space

60. What is the output of the recursive function if the input is 5 in the Fibonacci series?

a. 0, 1, 1, 2, 3, 5

b. 1, 1, 2, 3, 5, 8

c. 0, 1, 2, 3, 4, 5

d. 1, 2, 3, 4, 5, 6

Answer: b. 1, 1, 2, 3, 5, 8

61. What is dynamic memory allocation in C?

a. Allocating memory during runtime

b. Allocating memory at compile-time

c. Allocating memory for static variables

d. Allocating memory for global variables

Answer: a. Allocating memory during runtime

62. Which function is used to dynamically allocate memory in C?

a. malloc()

b. alloc()

c. memalloc()

d. allocate()

Answer: a. malloc()

63. What is the purpose of the free() function in C dynamic memory allocation?

a. Allocating memory

b. Freeing up allocated memory

c. Checking memory availability

d. Initializing memory

Answer: b. Freeing up allocated memory

64. What happens if you forget to free dynamically allocated memory in C?

a. Memory leak

b. Stack overflow

c. Program crash

d. Buffer overflow

Answer: a. Memory leak

65. Which of the following functions is used to reallocate memory in C?

a. realloc()

b. resize()

c. memresize()

d. allocate()

Answer: a. realloc()

66. What is the purpose of the calloc() function in C dynamic memory allocation?

a. Allocating memory for a single variable

b. Allocating contiguous memory blocks

c. Initializing memory to zero

d. Freeing up memory

Answer: c. Initializing memory to zero

67. How is memory deallocated in C after using malloc()?

a. Using free()

b. Using delete

c. Automatically by the system

d. Using dealloc()

Answer: a. Using free()

68. What does the sizeof() operator do in C?

a. Returns the size of a data type

b. Returns the size of allocated memory

c. Returns the size of the program

d. Returns the size of the hard disk

Answer: a. Returns the size of a data type

69. What is a memory block in the context of dynamic memory allocation?

a. A block of code

b. A block of stack memory

c. A block of heap memory

d. A block of global memory

Answer: c. A block of heap memory

70. What is the purpose of the memset() function in C dynamic memory allocation?

a. Allocating memory

b. Freeing up memory

c. Initializing memory to a specific value

d. Reallocating memory

Answer: c. Initializing memory to a specific value

71. How is dynamic memory allocated for an array of integers in C using malloc()?

a. intArray = malloc(sizeof(int))

b. intArray = malloc(sizeof(int) * n)

c. intArray = malloc(sizeof(intArray))

d. intArray = malloc(n)

Answer: b. intArray = malloc(sizeof(int) * n)

72. Which function is used to allocate memory for an array of structures in C?

a. malloc()

b. calloc()

c. realloc()

d. allocate()

Answer: a. malloc()

73. What is the purpose of the realloc() function in C dynamic memory allocation?

a. Allocating memory

b. Freeing up memory

c. Reallocating or resizing memory

d. Initializing memory

Answer: c. Reallocating or resizing memory

74. How is dynamic memory allocated for a string in C using malloc()?

a. stringVar = malloc(sizeof(char))

b. stringVar = malloc(strlen(stringVar))

c. stringVar = malloc(strlen(stringVar) + 1)

d. stringVar = malloc(sizeof(stringVar))

Answer: c. stringVar = malloc(strlen(stringVar) + 1)

75. Which function is used to free memory allocated for a dynamically allocated array in C?

a. dealloc()

b. remove()

c. free()

d. delete()

Answer: c. free()

76. What is the quadratic formula used for?

a. Finding the roots of a linear equation

b. Finding the roots of a quadratic equation

c. Sorting elements in an array

d. Searching for an element in an array

Answer: b. Finding the roots of a quadratic equation

77. In the quadratic equation ax^2 + bx + c = 0, what does 'a' represent?

a. Coefficient of x^2

b. Coefficient of x

c. Constant term

d. Variable term

Answer: a. Coefficient of x^2

78. What is the discriminant in the quadratic formula used for?

a. Determining the nature of roots

b. Finding the maximum value

c. Sorting elements

d. Searching for an element in an array

Answer: a. Determining the nature of roots

79. How many roots does a quadratic equation have if the discriminant is negative?

a. One real root

b. Two real roots

c. Two complex roots

d. No real roots

Answer: c. Two complex roots

80. What is the purpose of the minimum and maximum algorithms?

a. Sorting elements

b. Finding the range of elements

c. Finding the minimum and maximum values in a set

d. Searching for an element in an array

Answer: c. Finding the minimum and maximum values in a set

81. In the context of finding minimum and maximum numbers, what is a naive approach?

a. A simple and straightforward approach

b. A complex algorithm

c. A recursive approach

d. A random approach

Answer: a. A simple and straightforward approach

82. What is the significance of checking for a prime number in algorithms?

a. Determining the complexity

b. Avoiding redundant calculations

c. Finding the minimum value

d. Sorting elements

Answer: b. Avoiding redundant calculations

83. How do you check if a number is prime?

a. Count its factors

b. Use the Sieve of Eratosthenes

c. Check divisibility by numbers less than its square root

d. Use the Euclidean algorithm

Answer: c. Check divisibility by numbers less than its square root

84. In the linear search algorithm, how is the array traversed?

a. Randomly

b. Backward

c. Forward, one element at a time

d. In reverse order

Answer: c. Forward, one element at a time

85. What is the time complexity of the linear search algorithm in the worst case?

a. O(log n)

b. O(n)

c. O(n^2)

d. O(1)

Answer: b. O(n)

86. How does the binary search algorithm work?

a. Divides the array into two halves and compares the target element with the middle element

b. Compares each element with the target element linearly

c. Sorts the array first and then searches for the target element

d. Uses recursion to search for the target element

Answer: a. Divides the array into two halves and compares the target element with the middle element

87. What is the time complexity of the binary search algorithm in the worst case?

    a. O(log n)

    b. O(n)

    c. O(n^2)

    d. O(1)

    Answer: a. O(log n)

88. What is a key characteristic of the Bubble Sort algorithm?

    a. Stable sorting algorithm

    b. In-place sorting algorithm

    c. Recursive sorting algorithm

    d. Linear sorting algorithm

    Answer: b. In-place sorting algorithm

89. How does the Bubble Sort algorithm work?

    a. Divides the array into two halves

    b. Repeatedly swaps adjacent elements if they are in the wrong order

    c. Selects the minimum element and moves it to the front

    d. Uses recursion to sort the array

    Answer: b. Repeatedly swaps adjacent elements if they are in the wrong order

90. What is the time complexity of the Bubble Sort algorithm in the worst case?

    a. O(log n)

    b. O(n)

    c. O(n^2)

    d. O(1)

    Answer: c. O(n^2)

91. What is a key characteristic of the Insertion Sort algorithm?

    a. Stable sorting algorithm

    b. In-place sorting algorithm

    c. Recursive sorting algorithm

    d. Linear sorting algorithm

    Answer: b. In-place sorting algorithm

92. How does the Insertion Sort algorithm work?

    a. Divides the array into two halves

    b. Repeatedly selects the minimum element and moves it to the front

    c. Repeatedly takes one element from the unsorted part and inserts it into its correct position in the sorted part

    d. Uses recursion to sort the array

    Answer: c. Repeatedly takes one element from the unsorted part and inserts it into its correct position in the sorted part

93. What is the time complexity of the Insertion Sort algorithm in the worst case?

    a. O(log n)

    b. O(n)

    c. O(n^2)

    d. O(1)

    Answer: c. O(n^2)

94. What is a key characteristic of the Selection Sort algorithm?

    a. Stable sorting algorithm

    b. In-place sorting algorithm

    c. Recursive sorting algorithm

    d. Linear sorting algorithm

Answer: b. In-place sorting algorithm

95. How does the Selection Sort algorithm work?

    a. Divides the array into two halves

    b. Repeatedly swaps adjacent elements if they are in the wrong order

    c. Repeatedly selects the minimum element and moves it to the front

    d. Uses recursion to sort the array

    Answer: c. Repeatedly selects the minimum element and moves it to the front

96. What is the time complexity of the Selection Sort algorithm in the worst case?

    a. O(log n)

    b. O(n)

    c. O(n^2)

    d. O(1)

    Answer: c. O(n^2)

97. What is the order of complexity of an algorithm?

    a. The complexity of a sorting algorithm

    b. The efficiency of an algorithm

    c. The time it takes to execute an algorithm

    d. A mathematical function representing the relationship between the size of the input and the number of operations

    Answer: d. A mathematical function representing the relationship between the size of the input and the number of operations

98. What does O(n) represent in the context of the order of complexity?

    a. Constant time complexity

    b. Linear time complexity

    c. Quadratic time complexity

d. Exponential time complexity

Answer: b. Linear time complexity

99. Which of the following order of complexity is the most efficient?

a. O(log n)

b. O(n)

c. O(n^2)

d. O(2^n)

Answer: a. O(log n)

100. What is the purpose of Big O notation in algorithm analysis?

a. Representing the best-case scenario

b. Representing the worst-case scenario

c. Representing the average-case scenario

d. Representing all cases of an algorithm's performance

Answer: b. Representing the worst-case scenario

101. What is the purpose of the Order of Complexity in algorithm analysis?

a. Representing the best-case scenario

b. Representing the worst-case scenario

c. Representing the average-case scenario

d. Representing all cases of an algorithm's performance

Answer: d. Representing all cases of an algorithm's performance

102. What does O(n log n) represent in the context of the order of complexity?

a. Linear time complexity

b. Quadratic time complexity

c. Logarithmic time complexity

d. Exponential time complexity

Answer: c. Logarithmic time complexity

103. Which sorting algorithm typically has the best average-case time complexity?

a. Bubble Sort

b. Insertion Sort

c. Selection Sort

d. Merge Sort

Answer: d. Merge Sort

104. What is the purpose of the Merge Sort algorithm?

a. Repeatedly swaps adjacent elements

b. Repeatedly selects the minimum element and moves it to the front

c. Divides the array into two halves, sorts each half, and then merges them

d. Repeatedly takes one element from the unsorted part and inserts it into its correct position in the sorted part

Answer: c. Divides the array into two halves, sorts each half, and then merges them

105. How does the Merge Sort algorithm achieve better time complexity than Bubble, Insertion, and Selection sorts?

a. By using a divide-and-conquer approach

b. By using a recursive approach

c. By using a linear approach

d. By using an in-place approach

Answer: a. By using a divide-and-conquer approach

106. What is the time complexity of the Merge Sort algorithm in the worst case?

a. O(log n)

b. O(n)

c. O(n log n)

d. O(n^2)

Answer: c. O(n log n)

107. Which sorting algorithm is known for its stability, making it suitable for sorting records with multiple keys?

a. Bubble Sort

b. Insertion Sort

c. Selection Sort

d. Merge Sort

Answer: d. Merge Sort

108. What does it mean for a sorting algorithm to be stable?

a. It produces the same output every time it runs.

b. It maintains the relative order of equal elements in the sorted output as they appeared in the input.

c. It has a constant time complexity.

d. It can handle large datasets efficiently.

Answer: b. It maintains the relative order of equal elements in the sorted output as they appeared in the input.

109. What is the primary disadvantage of the Bubble Sort algorithm?

a. It is not stable.

b. It has a high time complexity.

c. It requires additional space.

d. It is not an in-place sorting algorithm.

Answer: b. It has a high time complexity.

110. In the context of sorting algorithms, what is an "in-place" algorithm?

    a. An algorithm that uses extra space for sorting.

    b. An algorithm that modifies the input data directly without requiring additional space.

    c. An algorithm that only works for small datasets.

    d. An algorithm that uses recursion.

    Answer: b. An algorithm that modifies the input data directly without requiring additional space.

111. Which of the following sorting algorithms is not an in-place sorting algorithm?

    a. Bubble Sort

    b. Insertion Sort

    c. Selection Sort

    d. Merge Sort

    Answer: d. Merge Sort

112. What is the primary advantage of the Insertion Sort algorithm?

    a. It has a low time complexity.

    b. It is an in-place sorting algorithm.

    c. It is stable and suitable for small datasets or partially ordered datasets.

    d. It is resistant to large datasets.

    Answer: c. It is stable and suitable for small datasets or partially ordered datasets.

113. What is the primary disadvantage of the Insertion Sort algorithm?

    a. It has a high time complexity.

    b. It is not a stable sorting algorithm.

    c. It is not an in-place sorting algorithm.

    d. It is not suitable for small datasets.

Answer: a. It has a high time complexity.

114. What is the primary advantage of the Selection Sort algorithm?

    a. It is stable and suitable for small datasets.

    b. It has a low time complexity.

    c. It is an in-place sorting algorithm.

    d. It is resistant to large datasets.

    Answer: c. It is an in-place sorting algorithm.

115. What is the primary disadvantage of the Selection Sort algorithm?

    a. It has a high time complexity.

    b. It is not a stable sorting algorithm.

    c. It is not an in-place sorting algorithm.

    d. It is not suitable for small datasets.

    Answer: b. It is not a stable sorting algorithm.

    Explanation: Selection Sort may change the relative order of equal elements, making it not stable.

116. What is the primary advantage of the Merge Sort algorithm?

    a. It has a low time complexity.

    b. It is an in-place sorting algorithm.

    c. It is stable and suitable for small datasets.

    d. It is resistant to large datasets.

    Answer: d. It is resistant to large datasets.

117. What is the primary disadvantage of the Merge Sort algorithm?

    a. It has a high time complexity.

    b. It is not a stable sorting algorithm.

c. It is not an in-place sorting algorithm.

d. It requires additional space for merging.

Answer: c. It is not an in-place sorting algorithm.

118. Which of the following statements is true about the time complexity of sorting algorithms?

a. The best-case time complexity is always more important than the worst-case time complexity.

b. The worst-case time complexity is always more important than the best-case time complexity.

c. Both the best-case and worst-case time complexities are important, depending on the specific use case.

d. Time complexity does not matter in sorting algorithms.

Answer: c. Both the best-case and worst-case time complexities are important, depending on the specific use case.

119. What is the purpose of the quicksort algorithm?

a. Sorting elements using a divide-and-conquer approach.

b. Inserting elements into a sorted array.

c. Selecting the minimum element and moving it to the front.

d. Merging two sorted arrays.

Answer: a. Sorting elements using a divide-and-conquer approach.

120. How does the quicksort algorithm work?

a. Repeatedly swaps adjacent elements if they are in the wrong order.

b. Repeatedly selects the minimum element and moves it to the front.

c. Divides the array into two partitions, sorts each partition, and then combines them.

d. Repeatedly takes one element from the unsorted part and inserts it into its correct position in the sorted part.

Answer: c. Divides the array into two partitions, sorts each partition, and then combines them.

121. What is the time complexity of the quicksort algorithm in the average case?

    a. O(log n)

    b. O(n)

    c. O(n log n)

    d. O(n^2)

    Answer: c. O(n log n)

122. Which of the following statements about quicksort is true?

    a. Quicksort is always a stable sorting algorithm.

    b. Quicksort has a linear time complexity in the worst case.

    c. Quicksort can be implemented as an in-place sorting algorithm.

    d. Quicksort is not suitable for large datasets.

    Answer: c. Quicksort can be implemented as an in-place sorting algorithm.

123. What is a hash function used for in algorithms?

    a. Sorting elements in an array.

    b. Finding the minimum and maximum values in a set.

    c. Mapping data to a fixed-size array, typically for fast data retrieval.

    d. Checking for prime numbers.

    Answer: c. Mapping data to a fixed-size array, typically for fast data retrieval.

124. In the context of hash functions, what is a collision?

    a. When two elements in an array are compared.

    b. When the hash function produces the same index for two different keys.

    c. When searching for an element in an array.

d. When sorting elements in an array.

Answer: b. When the hash function produces the same index for two different keys.

125. How does linear probing handle collisions in hash tables?

a. It creates a linked list at each index.

b. It moves to the next available index if a collision occurs.

c. It rehashes the entire table.

d. It ignores collisions and overwrites existing values.

Answer: b. It moves to the next available index if a collision occurs.