QUESTIONS AND ANSWERS

Sercan Külcü | 21.06.2023

Contents

1 Question and Answers

Explain the difference between the preprocessor directive #define and the const keyword in C. When should each be used?

What are function pointers in C? How are they declared and used? Provide an example scenario where function pointers can be useful.

In C, what is the difference between pass-by-value and pass-by-reference when passing arguments to a function? Explain with examples.

Discuss the concept of dynamic memory allocation in C. How is it different from static memory allocation? Provide examples of functions used for dynamic memory allocation.

Explain the purpose of the volatile keyword in C. When should it be used, and what potential issues does it address?

What are unions in C? How do they differ from structures? Provide a scenario where using a union would be appropriate.

Describe the concept of recursion in C programming. Provide an example of a recursive function and explain how recursion works step by step for that function.

What is the purpose of the restrict keyword in C? How does it affect the optimization of code and the compiler's ability to make certain optimizations?

What is the difference between malloc() and calloc() functions for dynamic memory allocation in C? When should each be used?

Explain the concept of file handling in C. Discuss the different file modes and provide an example of reading and writing data to a file using standard file handling functions.

Discuss the concept of function overloading in C. Can C support function overloading like C++? If not, how can similar functionality be achieved in C?

Explain the concept of multithreading in C. How can multithreading be implemented using native C libraries? Provide an example scenario where multithreading can improve program performance.

What is the purpose of the static keyword in C? Describe the differences between using static variables, functions, and global variables.

Discuss the concept of bit manipulation in C. Provide examples of bitwise operators and explain how they can be used for tasks such as setting, clearing, and toggling specific bits.

What are function prototypes in C? Why are they important, and how do they help in the compilation process? Provide an example demonstrating the use of function prototypes.

Explain the concept of function pointers to structures in C. How can function pointers be used to implement polymorphism in C? Provide a practical example.

Discuss the concept of inline functions in C. What are the advantages and disadvantages of using inline functions? When should inline functions be used?

What are the differences between automatic variables, static variables, and register variables in C? When should each type of variable be used, and what impact do they have on program execution?

Explain the concept of type casting in C. Discuss the different types of type casting operators and provide examples of explicit and implicit type casting.

Describe the concept of error handling in C. How can errors be detected and handled effectively in C programs? Discuss the use of error codes, exceptions, and error handling techniques.

Explain the concept of function pointers to arrays in C. How can function pointers be used to implement sorting algorithms such as bubble sort or quicksort?

Discuss the concept of macros in C. What are the advantages and disadvantages of using macros? Provide examples of how macros can be used effectively and any potential pitfalls.

Explain the concept of recursion versus iteration in C programming. When is it more appropriate to use recursion over iteration, and vice versa? Provide examples to illustrate your answer.

Discuss the concept of virtual memory in C programming. What is the role of virtual memory in memory management, and how does it affect program execution? Explain the concept of paging and its relationship to virtual memory.

Explain the concept of function pointers to variadic functions in C. How can function pointers be used with functions that have a variable number of arguments? Provide an example scenario where variadic functions and function pointers can be used together.

Discuss the concept of constant pointers and pointer-to-constant in C. What is the difference between const int* and int* const? Provide examples to demonstrate their usage.

Explain the concept of signal handling in C. What are signals, and how can they be handled in C programs? Provide examples of common signals and how they can be intercepted and processed.

Discuss the concept of inline assembly in C programming. How can inline assembly code be used to access low-level features of the processor or perform specific operations that are not directly supported by C?

Explain the concept of data structures in C programming. Discuss the implementation and usage of linked lists, stacks, and queues. Provide examples of operations on these data structures.

Discuss the concept of function pointers to function pointers in C. How can nested function pointers be used to create more complex and flexible function call mechanisms? Provide examples to illustrate your answer.

In C programming, what is the difference between a shallow copy and a deep copy when it comes to copying arrays? Provide an example scenario where a deep copy would be necessary.

What is function pointer casting in C? Explain how it can be used to achieve polymorphism in C.

In C, what are volatile variables and how are they different from regular variables? Provide a real-life example where using the volatile keyword would be necessary.

Can you explain the concept of bit manipulation in C? Provide an example of how bitwise operators can be used to perform complex operations efficiently.

What are the advantages and disadvantages of using recursion in C programming? Provide an example where recursion is a better approach compared to iteration.

Explain the concept of function pointers and how they can be used to implement callback functions in C programming. Provide an example scenario where callback functions are useful.

What are variadic functions in C? How are they implemented and when would you use them in a program?

Describe the purpose and usage of the volatile and const keywords in C programming. Provide examples to illustrate their significance.

In C, what are unions and how are they different from structures? Provide an example where a union would be more appropriate to use than a structure.

Explain the concept of dynamic memory allocation in C. Discuss the differences between malloc(), calloc(), and realloc(), and provide examples of when each function should be used.