Table of Contents

Q&A Top 50 in C
Q&A on Header files in C
Q&A on main function in C
Q&A on comments in C
Q&A on variables in C

Q&A Top 50 In C

1. What is C programming language?
   C is a high-level programming language that was developed in the early 1970s. It is widely used for system programming and is known for its efficiency and low-level programming capabilities.
2. What is the difference between a compiler and an interpreter?
   A compiler translates the entire source code into machine code before execution, while an interpreter translates and executes the source code line by line.
3. What is the difference between ‘char’ and ‘unsigned char’ data types in C?
   The ‘char’ data type can hold both positive and negative values, while the ‘unsigned char’ data type can hold only positive values.
4. What is the difference between ‘calloc’ and ‘malloc’?
   ‘malloc’ is used to allocate memory dynamically, while ‘calloc’ is used to allocate memory dynamically and initializes it to zero.
5. What is a pointer in C?
   A pointer is a variable that holds the memory address of another variable. It allows direct manipulation of memory locations.
6. What is the ‘const’ keyword used for in C?
   The ‘const’ keyword is used to declare variables as constants, meaning their values

cannot be modified once assigned.

7. What is the purpose of the ‘volatile’ keyword in C?
   The ‘volatile’ keyword is used to indicate that a variable can be modified by external entities, preventing certain compiler optimizations.
8. What is the purpose of the ‘static’ keyword in C?
   The ‘static’ keyword has different meanings depending on the context. It can be used to limit the scope of a variable to a particular file or to make a variable retain its value between function calls.
9. What is the difference between ‘printf’ and ‘sprintf’ functions in C?
   ‘printf’ prints formatted output to the standard output stream (console), while ‘sprintf’ prints formatted output to a string.
10. What is the difference between ‘strcpy’ and ‘strncpy’ functions in C?
    ‘strcpy’ copies the content of one string to another until it encounters a null character, while ‘strncpy’ copies a specified number of characters, and if the null character is not encountered, it appends one to the destination string.
11. What is the purpose of the ‘sizeof’ operator in C?
    The ‘sizeof’ operator is used to determine the size in bytes of a variable or a data type.
12. What is the difference between ‘++i’ and ‘i++’?
    ‘++i’ is the pre-increment operator that increments the value of ‘i’ and then returns the incremented value. ‘i++’ is the post-increment operator that returns the value of ‘i’ and then increments it.
13. What is the difference between ‘while’ and ‘do-while’ loops?
    A ‘while’ loop tests the condition before executing the loop, while a ‘do-while’ loop executes the loop at least once before testing the condition.
14. What is recursion?
    Recursion is a programming technique where a function calls itself directly or indirectly.

15. What is the difference between global variables and local variables?
    Global variables are defined outside any function and have a global scope, while local variables are defined within a function and have a local scope.
16. Explain the ‘typedef’ keyword in C.
    ‘typedef’ is used to create aliases for existing data types. It allows you to define custom names for existing types, making code more readable and portable.
17. What is a structure in C?
    A structure is a user-defined data type that allows combining different types of variables under a single name.
18. What is a union in C?
    A union is a user-defined data type that allows storing different types of variables at the same memory location.
19. What is the purpose of ‘enum’ in C?
    ‘enum’ is used to define a set of named constants, making the code more readable and maintainable.
20. What is the difference between ‘NULL’ and ‘nullptr’ in C?
    ‘NULL’ is a macro defined in C that represents a null pointer, while ‘nullptr’ is a keyword introduced in C++11 that represents a null pointer.
21. What is a file pointer in C?
    A file pointer is a pointer that is used to handle files in C. It points to the location of data in a file.
22. What is the ‘fopen’ function used for in C?
    ‘fopen’ is used to open a file. It returns a file pointer that is used for subsequent file operations.

23. What is the purpose of the ‘fwrite’ function in C?
    ‘fwrite’ is used to write binary data to a file.

24. What is the purpose of the ‘fread’ function in C?
    ‘fread’ is used to read binary data from a file.
25. What is the purpose of the ‘fprintf’ function in C?
    ‘fprintf’ is used to write formatted data to a file.

26. What is the difference between ‘fputs’ and ‘puts’ functions in C?
    ‘fputs’ writes a string to a file, while ‘puts’ writes a string to the standard output.
27. What is the purpose of the ‘feof’ function in C?
    ‘feof’ is used to check if the end of a file has been reached.
28. What is the purpose of the ‘fgets’ function in C?
    ‘fgets’ is used to read a line of text from a file.

29. What is the purpose of the ‘strcmp’ function in C?
    ‘strcmp’ is used to compare two strings. It returns 0 if the strings are equal.
30. What is the purpose of the ‘strlen’ function in C?
    ‘strlen’ is used to determine the length of a string.
31. What is the purpose of the ‘malloc’ function in C?
    ‘malloc’ is used to allocate memory dynamically.

32. What is the purpose of the ‘free’ function in C?
    ‘free’ is used to deallocate memory that was previously allocated using ‘malloc’ or ‘calloc’.
33. What is the difference between ‘==’, ‘>’, and ‘>=’ operators?
    ‘==’ is the equality operator that checks if two values are equal. ‘>’ is the greater than operator that checks if the left operand is greater than the right operand. ‘>=’ is

the greater than or equal to operator that checks if the left operand is greater than or equal to the right operand.

34. What is the purpose of the ‘break’ statement in C?
    The ‘break’ statement is used to exit from a loop or switch statement.
35. What is the purpose of the ‘continue’ statement in C?
    The ‘continue’ statement is used to skip the remaining statements in a loop and continue to the next iteration.
36. What is the difference between ‘struct’ and ‘typedef struct’ in C?
    ‘struct’ is used to define a structure without creating a new type name, while ‘typedef struct’ is used to define a structure and create a new type name simultaneously.
37. What is the purpose of the ‘sizeof’ operator when used with a structure or union?
    The ‘sizeof’ operator returns the size in bytes of a structure or union.
38. What is a function pointer in C?
    A function pointer is a pointer that holds the address of a function. It allows dynamic selection and invocation of functions at runtime.
39. What is the purpose of the ‘void’ keyword in C?
    The ‘void’ keyword is used to indicate the absence of a specific type. It is used for functions that do not return a value or as a parameter type for functions that do not accept any arguments.
40. What is a header file in C?
    A header file is a file that contains declarations of functions, variables, and data types. It is included in a C program using the ‘#include’ directive.
41. What is the purpose of the ‘assert’ macro in C?
    The ‘assert’ macro is used for debugging purposes. It checks if a given expression is true and terminates the program if it is false.
42. What is the purpose of the ‘static’ keyword when used with a global variable?
    The ‘static’ keyword limits the scope of a global variable to the file in which it is

defined. It cannot be accessed from other files.

43. What is a command-line argument in C?
    A command-line argument is an input parameter that is passed to a C program when it is executed from the command line.

44. What is the purpose of the ‘getchar’ function in C?
    ‘getchar’ is used to read a single character from the standard input.
45. What is the purpose of the ‘putchar’ function in C?
    ‘putchar’ is used to write a single character to the standard output.
46. What is the purpose of the ‘strupr’ function in C?
    ‘strupr’ is used to convert a string to uppercase.
47. What is the purpose of the ‘tolower’ function in C?
    ‘tolower’ is used to convert a character to lowercase.

48. What is a bitwise operator in C?
    Bitwise operators perform operations at the bit level. They include AND (‘&’), OR (‘|’), XOR (‘^’), left shift (‘<<‘), and right shift (‘>>’).
49. What is the purpose of the ‘const’ qualifier when used with a function parameter?
    The ‘const’ qualifier indicates that the function will not modify the value of the parameter.
50. What is the difference between ‘rand’ and ‘srand’ functions in C?
    ‘rand’ generates a pseudo-random integer, while ‘srand’ seeds the random number generator used by ‘rand’.

Q&A On Header Files In C

Q1. What is a header file in C?

Ans. A header file in C is a file containing function prototypes, type definitions, macro definitions, and other declarations that can be included in multiple source files. It provides a way to share common declarations across multiple files.

Q2. How do you include a header file in C?

Ans. The #include directive is used to include a header file in C. It allows the contents of the header file to be copied and pasted into the source file at the location of the #include statement.
Example usage:

```
#include <stdio.h>
```

Q3. What is the difference between and “header.h” in an #include statement?

- <header.h> is used to include system header files. The compiler searches for these header files in the standard system directories.
- “header.h” is used to include user-defined header files. The compiler searches for

these header files in the current directory or the directories specified by the -I flag.

Q4. What are the advantages of using header files?

- Encapsulation: Header files allow you to separate the interface (function prototypes, declarations) from the implementation (source code), promoting encapsulation and modularity.
- Code Reusability: Header files enable you to reuse code across multiple source files by providing a centralized location for common declarations.
- Readability and Maintainability: Header files make code more readable and maintainable by providing a clear overview of the functions, types, and macros used in a program.

Q5. Can you define a function in a header file?

Yes, you can define functions in a header file using the inline keyword. Defining functions in header files can lead to potential multiple definition errors if the header file is included in multiple source files without proper handling.

Example usage:

```
// math.h
#ifndef MATH_H
#define MATH_H

inline int square(int num) {
    return num * num;
}
```

```
#endif
```

Q&A On Main Function In C

Q6. What is the purpose of the main function in C?

The main function is the entry point of a C program. It is called automatically when the program starts execution. The main function is responsible for executing the program's logic and typically contains the program's statements and function calls.

Q7. What is the return type of the main function?

The return type of the main function is int. It indicates the status of the program's execution to the operating system. By convention, a return value of 0 typically indicates successful execution, while non-zero values indicate an error or abnormal termination.

Q8. Can the main function be called recursively?

Yes, technically, the main function can be called recursively like any other function in C. However, it is generally not recommended to call main recursively because it can lead to confusion and may result in unexpected behavior. The preferred way to reuse code is by using functions other than main.

Q9.What are the command-line arguments of the main function?

- The main function can accept two parameters: argc (argument count) and argv (argument vector). They allow you to pass command-line arguments to the program

when it is executed.

- argc represents the number of command-line arguments passed to the program.
- argv is an array of strings (character arrays) that contain the actual command-line arguments.

Q10. How do you access command-line arguments in the main function?

Command-line arguments can be accessed through the argc and argv parameters of the main function. The first command-line argument (argv[0]) is always the name of the program itself.

Example usage:

```
int main(int argc, char *argv[]) {
    printf("Number of arguments: %d\n", argc);
    for (int i = 0; i < argc; i++) {
        printf("Argument %d: %s\n", i, argv[i]);
    }
    return 0;
}
```

Q&A On Comments In C

Q11. What is a comment in C?

A comment in C is a piece of text that is ignored by the compiler. It is used to add explanatory notes or annotations to the code, making it easier to understand and maintain.

Q12. What are the two types of comments in C?

There are two types of comments in C:

- Single-line comments: These comments start with // and continue until the end of the line. They are used to comment on a single line of code.

```
// This is a single-line comment
```

- Multi-line comments: These comments start with /* and end with */. They can span multiple lines and are used to comment on a block of code.

```
/*
This is a
multi-line comment
*/
```

Q13. What is the purpose of comments in a program?

Comments serve several purposes in a program:

- Documentation: Comments help explain the purpose and functionality of code, making it easier for developers to understand and maintain.
- Debugging: Comments can be used to temporarily disable code for debugging purposes without removing it completely.
- Collaboration: Comments aid in collaboration between team members by providing

additional context and explanations for the code.
- Readability: Well-placed comments enhance the readability of code by providing high-level overviews or clarifying complex logic.

Q14. Can comments be nested in C?

No, comments cannot be nested in C. If you try to nest one comment within another, it will result in a compilation error.

Q15. Can comments be used within preprocessor directives?

Yes, comments can be used within preprocessor directives. However, comments within preprocessor directives are treated differently by the preprocessor and are not included in the final processed code.

Q&A On Variables In C

Q16. What is a variable in C?

A variable in C is a named location in memory that stores a value. It has a specific data type (such as int, float, char, etc.) that determines the type of data it can hold.

Q17. How do you declare a variable in C?

In C, variables are declared by specifying the data type followed by the variable name.

Example:

```
int age;
float salary;
char grade;
```

Q18. What is the scope of a variable in C?

The scope of a variable determines the region of the program where the variable is visible and can be accessed.
In C, variables can have one of three scopes:

- Local scope: Variables declared within a block of code (e.g., inside a function) have local scope. They are accessible only within that block.
- Global scope: Variables declared outside of any function have global scope. They are accessible from any part of the program.
- Function parameter scope: Variables declared as function parameters have the scope of that function.

Q19. What is the difference between automatic and static variables?

- Automatic variables are declared inside a block of code and have a local scope. They are created when the block is entered and destroyed when the block is exited. Automatic variables are not explicitly initialized and their values are undefined unless assigned.
- Static variables are also declared inside a block of code but retain their value even after the block is exited. They have a local scope but are initialized only once and retain their value across function calls.

Q20. Can you change the value of a constant variable?

No, constant variables (declared using the const keyword) cannot be modified once they are assigned a value. Attempting to modify the value of a constant variable will result in a compilation error.

Related posts:

1. Machine Learning Interview Q&A
2. Python Interview Q&A
3. Top PHP Interview Questions and Answers for Success
4. C++ Programming Interview Q&A
5. Java Interview Q&A
6. Jupyter Notebook Inteview Q&A
7. Computer Networks Interview Q&A
8. C# Q and A
9. Android App Deveopment Q&A
10. R Interview Q&A
11. HTML Interview Q&A
12. Basic computer interview Q&A
13. Data Structure Interview Q&A
14. Vb Net top 50 interview questions and answers