**Programming Tips & Tricks**

1. **How to combine .h file and other .c files into a .exe file in C? [separating implementation from declaration]**

**Solution:** To combine a header file and multiple C source files into a single executable file, you will need to follow these steps:

1. Compile each C source file into an object file (.o) using a C compiler, such as GCC or Clang. For example, if you have two source files **main.c** and **functions.c**, you can compile them into object files with the following commands:

**gcc -c main.c**

**gcc -c functions.c**

1. Link the object files together using a linker, such as GCC or Clang, to create the executable file. For example, if you have two object files **main.o** and **functions.o**, you can link them together with the following command:

**gcc main.o functions.o -o myprogram**

This will create an executable file called **myprogram** in the current directory.

1. If you have a header file **myheader.h** that is included in both **main.c** and **functions.c**, you can include it in each C source file with the following line:

**#include "myheader.h"**

This will tell the compiler where to find the declarations of any functions or variables that are defined in the header file.

Note that if you have any external libraries that your program depends on, you will need to link them in as well. You can do this by adding the appropriate linker flags, such as **-lm** for the math library, to the linker command.

1. **#ifndef\_\_SINGLY\_LIST\_\_ ; #define\_\_SINGLY\_LIST\_\_ #endif;**

The #ifndef and #define directives are used in C and C++ to create header guards. Header guards are used to prevent a header file from being included multiple times in the same source file, which can cause errors due to redefinition of functions or variables.

The #ifndef directive checks whether the symbol \_\_SINGLY\_LIST\_\_ has been defined. If it has not been defined, the code inside the #ifndef block will be executed. The #define directive then defines the symbol \_\_SINGLY\_LIST\_\_. This symbol can be any name, as long as it is unique to the header file.

If the header file is included multiple times in a source file, the #ifndef block will be skipped the second time the file is included, because the symbol \_\_SINGLY\_LIST\_\_ will already have been defined. This ensures that the code in the header file is only compiled once, avoiding errors due to duplicate definitions.

Here's an example of how header guards can be used:

**#ifndef MYHEADER\_H**

**#define MYHEADER\_H**

**/\* code goes here \*/**

**#endif**

**3. Difference between “malloc” and “calloc” in C.**

malloc and calloc are both functions used for dynamic memory allocation in C programming language. They differ in the way they allocate and initialize the memory.

1. malloc: malloc stands for memory allocation. It allocates a block of memory of a specified size and returns a pointer to the first byte of the block. The memory block allocated by malloc is not initialized and may contain garbage values. The syntax of malloc is as follows:  
     
    ptr = (cast-type\*)malloc(byte-size);
2. calloc: calloc stands for contiguous allocation. It also allocates a block of memory of a specified size, but it initializes the memory block to zero. The syntax of calloc is as follows:  
     
    ptr = (cast-type\*)calloc(num-elements,element-size);

Here, num-elements specifies the number of elements to be allocated, and element-size specifies the size of each element. calloc returns a pointer to the first byte of the allocated memory block.