3) Class definitions-- Classes let you create objects. They are essentially new data types that may contain functions for retrieving, setting or manipulating the objects to which they belong. This will set up much of the functionality for the rest of your program.

4) Member function definitions-- Classes usually contain functions, most of which will be defined outside of the actual class definition. Member functions are defined outside of the class definition brackets not only to keep your program better organized, but also because functions defined within the brackets are inline functions by default, which you may want to avoid (or at least let the compiler make decisions about).

5) Other function declarations -- Functions not particular to any class and not #included in a header file are declared next. You may also define them at this point, but most functions are simply declared; that is, without coding the body of the function. Such functions are known as "global" functions because, like global variables, they are visible everywhere in the program.

6) Global variable declarations-- Variables that will be visible to all parts of your program should be declared before `main()`.

7) `main()`-- As in the "Hello World" example, `main()` must appear in every program. This is where it all starts.

8) Function definitions -- It is your choice whether or not to define your functions when you declare them. Usually, functions longer than 1 or 2 lines are defined below `main()` for easier readability.

Problems

- **Problem :** What is the purpose of the #include statement in a C++ program?

The `#include` statement tells the C++ preprocessor to insert the definitions of variables, classes, and functions into the current file.`#include <iostream.h>`, for example, lets you use the cout and cin functions, and `#include <math.h>` lets you use extra functions like the square root function `sqrt()`. To `#include` files that are