1. Why is a C/C++ program divided into multiple source files (modules), and why do they use distinctive filename extensions such as .cpp and .h?
   * C/C++ programs are divided into source files and modules for code reusability, to organize functionality, provide scope and extent of a variable or function, allows multiple programmers to work on the same code, among other reasons. Different file extensions are used because the purpose of each file is different: a header file (.h) is meant to include declarations of variables, functions, classes; and a .cpp file is meant to contain the implementation of such declarations.
2. What is the difference between a pointer and reference?

A reference must be initialized when declared while a pointer can be assigned a value later in the code. Also, the memory address of a pointer p can be accessed, while the memory address of a reference r cannot be accessed.

A reference is an “alias” of another variable: if r is a reference to x, then the address of r (&r) will be the same as the address of x (&x). However, if a pointer p is created and assigned the memory address of x, \*p points to the value of x, p will hold the address of x, but contrary to a reference, the address of the pointer p (&p) will not be the same as the address of the variable x (&x).

1. What is the function of "const" and what are some practical applications?
   * “const” is used to let know the compiler that the value of the variable being declared cannot change. One practical application is when defining numerical constants (i.e. the number “pi” or “e”) or a particular combination of numbers that cannot be modified while the program is running (i.e. social security). Having a “const” type qualifier also makes a code less prone to errors since the variable value cannot change.
2. Summarize what function overloading is and when to use it.
   * Function overloading is having 2 or more functions with the same name, but the number of parameters or the type of such parameters is not the same. We can use function overloading when we need to perform the same transformation to different data types.
3. Describe the difference between a *using* declaration and a *using* directive.
   * If there is code that will be used by multiple .cpp files, is best to use a directive to avoid errors. If there are functions, variables, a struct or any other c++ data type that is particular to one .cpp file, is best to declare the data type locally (and hence using declaration and not a directive). To use a directive in c++, we must use “#include <declaration>”. Then the code inside the declaration file gets added to the source code by the preprocessor, before compiling the code. A declaration is used to let the compiler know about functions, variables types, structs, or other c++ data types that will be used in the program.