Week13

a. What does the static keyword mean in front of a variable or function defined at the file scope?

在文件作用域(File Scope)即类或结构体之外定义的变量或函数前加上 `static` 关键字，意味着该符号具有**内部链接(Internal Linkage)**。

- **可见性(Visibility)：** 该变量或函数仅在定义它的**翻译单元(Translation Unit)**(即当前的 .cpp 文件)内部可见。
- **链接器行为(Linker Behavior)：** 链接器在链接阶段不会在该翻译单元之外查找该符号定义。这意味着你可以在不同的 .cpp 文件中定义同名的 `static` 变量，它们互不干扰，就像是在文件级别将其标记为“私有(Private)”一样。

b. How to declare and then use a file scope variable, which is defined in another .cpp file?

如果你需要访问在另一个翻译单元(.cpp 文件)中定义的全局变量，该变量首先不能被标记为 static(因为 static 会限制其为内部链接)。

在当前文件中，你需要使用 extern 关键字进行声明。这被称为外部链接(External Linkage)，它告诉链接器该符号存在于外部的某个翻译单元中，并在链接阶段进行解析。

c. What is the meaning of the static keyword when applied to a member variable or member function of a class or struct?

当 `static` 应用于类或结构体的成员时，其含义变为“共享”：

- **静态成员变量(Static Member Variable)：** 意味着该变量在类的所有**实例(Instances)*之间*共享内存(Share Memory)**。无论创建了多少个类的实例，该变量在内存中仅存在一份副本(One Instance)。
- **静态成员函数(Static Member Function)：** 意味着该函数不依赖于类的具体实例即可调用。静态成员函数内部没有当前类的实例上下文，因此无法访问非静态成员。

d. static member variables are only declared within a class or struct. Where should they be defined?

静态成员变量在类内部只是声明(Declared)。为了解决链接时的符号引用，必须在类外部(通常在 .cpp 文件中)的全局作用域进行定义(Defined)。

语法通常为：类型 类名::变量名;(例如 int Entity::x;)。如果不进行定义，链接器会报错“无法解析的外部符号(Unresolved External Symbol)”。

e. Why static member functions cannot access non-static member variables?

这是因为静态成员函数没有**隐式参数(Hidden Parameter)**——即 `this` 指针。

- **非静态成员函数(Non-static Member Function)：** 在编译时会隐式接收一个指向当前对象实例的指针(即 `this`)，从而能够定位和访问属于该特定实例的非静态成员变量。
- **静态成员函数(Static Member Function)：** 类似于类外部的普通函数，它不接收这个隐式实例指针。由于它不知道具体的实例是谁，因此无法访问依赖于具体对象实例的非静态数据。

f. What does the this keyword represent and refer to?

this 关键字是一个指针(Pointer)，它指向当前对象实例(Current Object Instance)。

它仅在非静态成员函数(Member Function)内部有效。通过 this 指针，函数可以访问和操作调用该函数的特定对象的成员变量。

g. How does the this keyword differ between a const member function and a non-const member function?

区别在于 `this` 指针的类型修饰：

- **非 const 成员函数(Non-const Member Function)：** `this` 的类型通常是 `Entity*`(指向 Entity 的指针)。你可以通过它修改成员变量。
- **const 成员函数(Const Member Function)：** `this` 的类型变为 `const Entity*`(指向 const Entity 的指针)。这意味着你不能通过该指针修改类的成员变量(即不能向其指向的内存写入数据)，从而保证了函数的只读属性(Read-only)。

h. What is the scope and extent of a local static variable?

局部静态变量(Local Static Variable)结合了局部作用域和全局生命周期：

- **生命周期(Lifetime)：** 它的生命周期贯穿整个程序的执行过程。它只在函数第一次被调用时初始化一次，之后其值在内存中一直保留，直到程序结束。
- **作用域(Scope)：** 它的作用域被限制在声明它的函数或代码块内部。虽然它在内存中一直存在，但除了该函数内部，其他任何代码都无法直接访问(Visible)它。