Search Task

-When should I use a struct instead of a class?

1-Struct:

-Struct is a value type, which means it is stored on the stack, and when you assign a struct to a new variable, a copy of the value is created.

-struct cannot be inherited from another struct or class, and it cannot serve as a base for other structs or classes. It can, however, implement interfaces.

-struct can be more memory-efficient in certain situations because it avoids the overhead associated with reference types (e.g., no garbage collection).

-Use struct for small, simple data structures that represent a single value or a collection of related values (e.g., Point, Rectangle, Color).

-Use struct When you need high performance and want to avoid the overhead of heap allocation and garbage collection.

2-Class:

-Class is a reference type, which means it is stored on the heap, and when you assign a class instance to a new variable, you are copying the reference to the same object, not the object itself.

-Class supports inheritance, meaning you can derive new classes from existing ones, allowing for polymorphism and code reuse.

-Instances of classes are managed by the garbage collector, which handles memory allocation and deallocation.

-Use class for complex data structures that involve behavior, inheritance, or polymorphism.

-Use classes When you need to create large objects that should not be copied frequently, as classes are more memory efficient for large data.