Static Scoping

static (lexical scoping): the bindings between names and objects can be determined at compile time by examining the text of the program, without consideration of the flow of control at runtime.

Typically in a language using **static scoping** the "current" binding for a given name is found in the matching declaration whose block most closely surrounds a given point in the program.

A **static** variable has a lifetime that encompasses the entire execution of the program. Instead of a logically separate object for every invocation of the subroutine, the compiler creates a signle object that retains its value from one invocation of the subroutine to the next.

In a language with static(lexical) scoping, the bindings between names and objects can be determined at compile time by examining the text of the program without consideration of the flow of control at run time. Typically, the "current" bidning for a given name is found in the matching declaration whose block most closely surrounds a given point in the program.

A static variable has a lifetime that encompasses the entire execution of the program. Instead of a logically separate object for every invocation of the subroutine, the compiler creates a single object that retains its value from one invocation of the subroutine to the next. (The name-to-variable binding, is inactive when the subroutine is not executing, because the name is out of scope.)