The Notion of Binding Time

Binding: A binding is an association between two things, such as a name and the thing it names.

Binding time: Binding time is the time at which a binding is created, or more generally, the time at which any **implementation** decision is made. Decisions may be bound at different times.

Language design time: in most languages, the control-flow constructs, the set of fundamental (primitive) types, the available constructs for creating complex types, and many other aspects of language semantics are chosen when the language is designed.

Language implementation time: Most language manualsleave a variety of issues to the discretion of the language implementor. Typically include the it precision (number of bits) of the fundamental *types*, the coupling of Î/O to the operating system's notation of *files*, and the organization and maximum sizes of the stack and heap.

Program writing time: Programmers choose algorithms, data structures, and names.

Compile time: Compilers choose the mapping og *high-level* constructs to machine code, including the layout of statically defined data in memory.

Link time: Most compilers support separate compilation— compiling different modules of a program at different times— and depend on the availability of a library of standard subroutines, a *program* is usually not complete until the various modules are joined together by a linker. The *linker* chooses the overall layout of the modules with respect to one another, and resolves intermodule references. When a name in one module refers to an object in another module, the binding between the two is not finalized until link time.

Load time: Load time refers to the point at which the operating system loads the program into memory so that it can run. Most modern *operating systems* distinguish between virtual addresses and physical addresses. *Virtual addresses* are chosen at link time; *physical addresses* can change at runtime. The processor's memory management hardware translates virtual addresses into physical addresses during each individual instruction at runtime

Runtime: covers the entire span from the beginning to the end of ececution. Bindings to values to variabes occur ar runtime. *Runtime* includes *program* start-up time. *Module* entry, elaboration time (the point at which a declaration is first "seen"), subroutine call time, block entry time, and expression evaluation, statement execution

static and dynamic are used to refer to before runtime and at runtime respectively.