names: strings of characters that stand for something

binding: association between two things
(e.g., name and the thing it names
generically, a question and its answer)
what happens on overflow?
what are the semantics of foo?

referencing environment: set of bindings currently in effect

binding time: time at which a binding is created

language design time (control constructs, primitive types)
language implementation time (C bits in int, float types)

static
\begin{align*}
\text{program writing} & \quad \text{(method names to source code)} \\
\text{compile time} & \quad \text{(methods to machine instructions)} \\
\text{link time} & \quad \text{(methods to virtual addresses)} \\
\text{load time} & \quad \text{(methods to physical addresses)} \\
\end{align*}

run time

\begin{align*}
\text{dynamic} & \\
\text{subroutine entry/exit (locals to address)} \\
\text{blocks entry/exit} \\
\text{start execution (variables to values)}
\end{align*}
lifetime: time between creation/destruction
(bindings and objects) \quad x = 1;

space allocation: static (global variables, constants, procedures)

stack (local variables)

heap (run objects, malloc/free, new/delete)

```c
char* hello();
char* strpy(char*, char*);

char* s = "Hello world";

char* str = strpy(s, "hello");

char* str1;

int main() {
    char* s2 = malloc(8);
    char* s3 = malloc(8);
    strcpy(s2, "hello");
    return 0;
}
```

A calls B, B calls C:

```
c(x, y, z) \quad \forall int a,
```
Stack grows dynamically local var name. Stack size is dynamic
name = stack offset is statically bound