5 Implementing Scope Rules

Monday, September 28, 2020 10:40 AM

Static Scoping:
When the scope of all names can be known by looking at the text of the program. I.e. at compile time. *aka* **Lexical Scoping**.

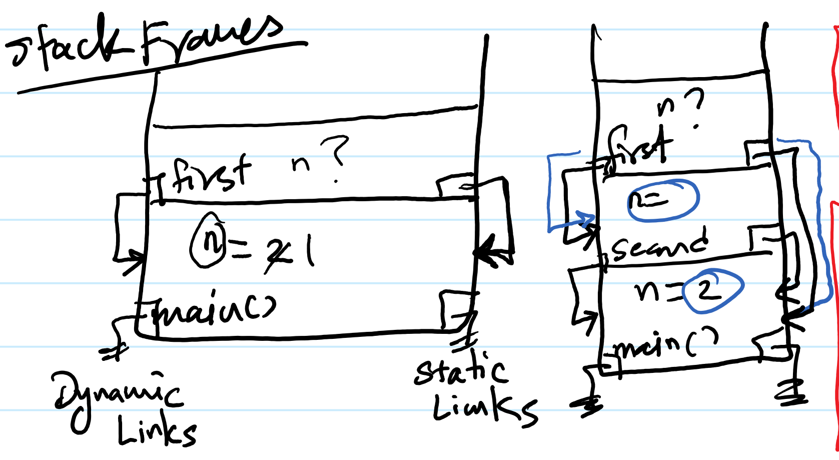
Dynamic scoping:
When the scope of a name can only be determined at run time, dictated by flow of execution of program.

	<=0	>0
Static	1	1
Dynamic	1	2

n = 2

```
int n

function first(…) {
    …
    n = 1
} // first()

function second(…) {
    int n
    first(…)
} // second()

n = 2;
if (read_integer() > 0)
    second(…)
else
    first()

print n
```

= 1

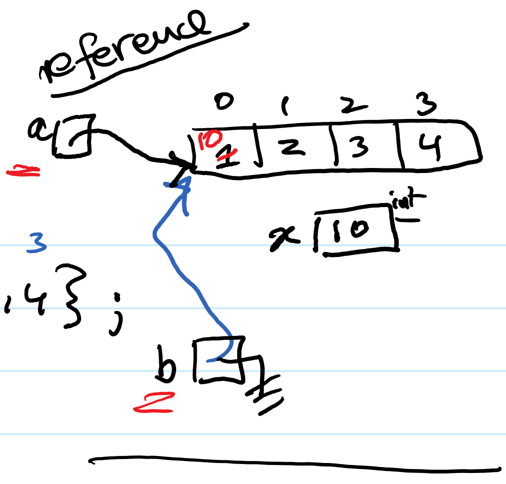


Referencing Environment:
Complete set of bindings at a given point in a program is called a **referencing environment**.

Can be determined by using stack frames and static/dynamic links. But there are some other issues…due to *aliasing*, *overloading*, *polymorphism*, *first-class values*, etc.

5 Aliasing, Overloading, Polymorphism, and First-class Values

Monday, September 28, 2020 9:00 AM



e.g. Java

```
int [] a = {1,2,3,4};
int x = 10;
int[] b;
b = a;

b[0] = 10;
print a[0] / a  →  ??     10, 2, 3, 4
```

Aliasing : when one or more names in a program refer to the same object at the same point in a program.

a & b are aliases

Common in all PLs where we have references/pointers.

eg. C++

```
void f (int &a, int & b) {
                         // a & b refer to the same object.
}
① f(x, x)
② f(l[i], l[j]);     i = j  ⇒ aliasing
```

C

```
int *n;              | void f (int *a){
void main(){         |    // a + n are aliases
   f(n);             | }
}
```

5 Aliasing, *contd*.

Monday, September 28, 2020 9:00 AM

Java + Python

5 Overloading: Operator and Function Overloading

Monday, September 28, 2020 9:00 AM

Overloading:
When a symbol in a program (PL) has more than one meaning.

Java

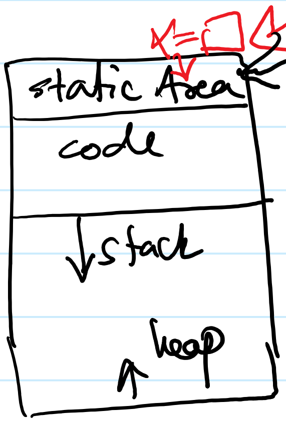
```
public class ← {
    public static int max (int a, int b) {
    }
}
```

2 diff. meanings !!

```
public static void f(——){
    static int x;
}
```

C

```
int void f(——){
    static int x;
}
```



static variables

Arithmetic operators $c = a + b;$ + is overloaded

Widget w1, w2, w3;

$w1 = w_2 + w_3;$