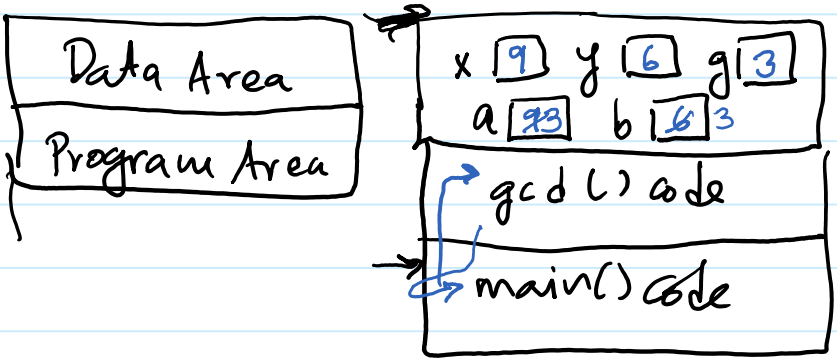
4 Static Allocation Scheme

Monday, September 21, 2020 5:10 PM

All allocation for **all data and code** is done at compile-time.
E.g. in pre-1990 Fortran.



Does not allow recursion!!

```
int gcd(int a, int b){
    if (a == b)
        return a;
    if (a > b)
        return gcd(a-b, b);
    else
        return gcd(a, b-a);
}
```

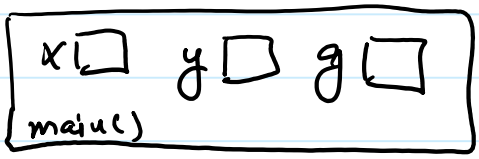
```
#include <stdio.h>

int gcd(int a, int b) {
    while (a != b) {
        if (a > b)
            a = a - b;
        else
            b = b - a;
    }
    return a;
} // gcd()

int main() {
    int x, y;

    // Input x and y

    int g = gcd(x, y);

    // Output g

    return 0;
} // main()
```

gcd(x, y)

4 Recursion with Static Allocation

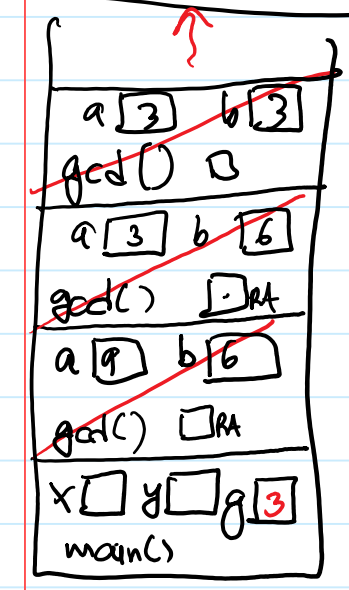
Monday, September 21, 2020 5:31 PM

Stack Frame | Activation Record



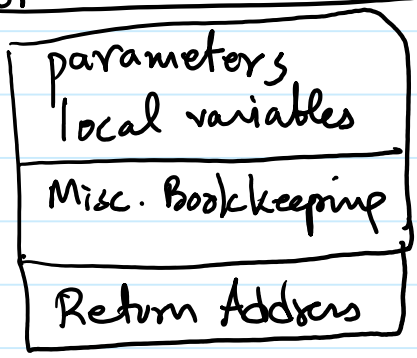

```
int gcd(int a, int b) {
    if (a == b)
        return a;
    else if (a > b)
        return gcd(a-b, b);
    else
        return gcd(a, b-a);
} // gcd()
```

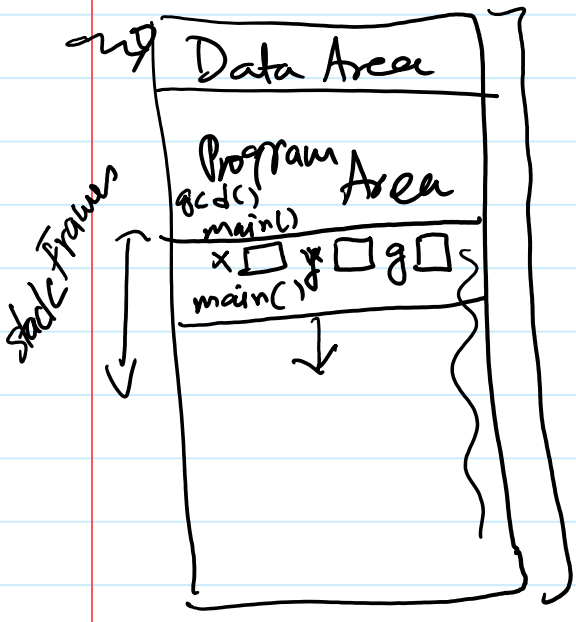
Run time Stack



gcd(9, 6)
a = 9, b = 6
3 → gcd(3, 6)
3 → gcd(3, 3) ⟵

Typical Stack Frame

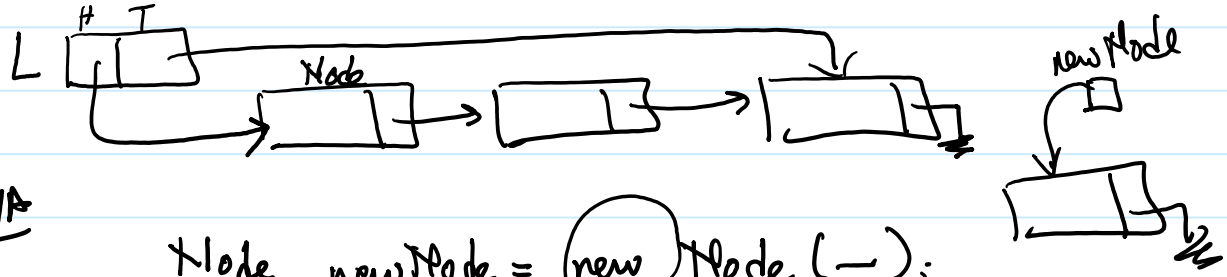




4 Heap-based Allocation

Monday, September 21, 2020 5:31 PM

for dynamically allocated data...



JAVA

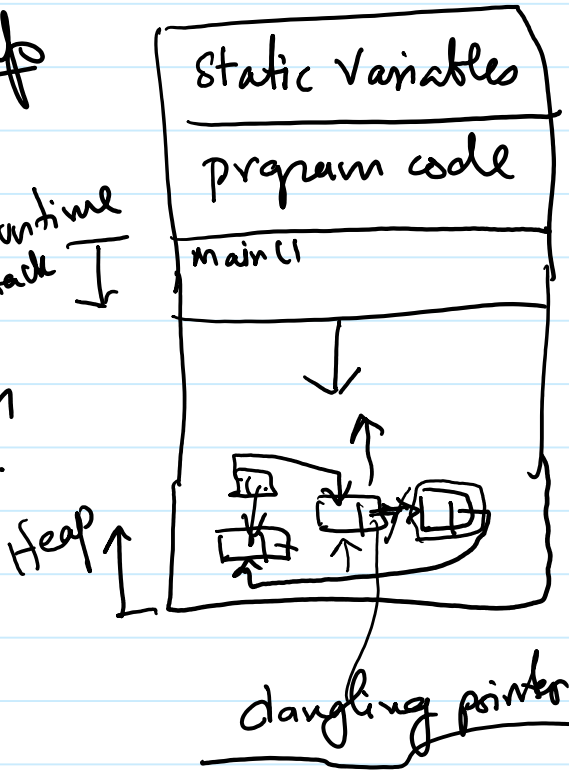
```
Node newNode = new Node(_);
```

C

```
Node *newNode = (Node) malloc(sizeof(Node));
```

Allocated on a **Heap** of bytes in memory!

Memory Map



Stack meets heap!

runtime stack

PL decision
who/when memory alloc & deallocation is done

* User's program instructions

free
c free
dispose

Garbage Collection
runtime process

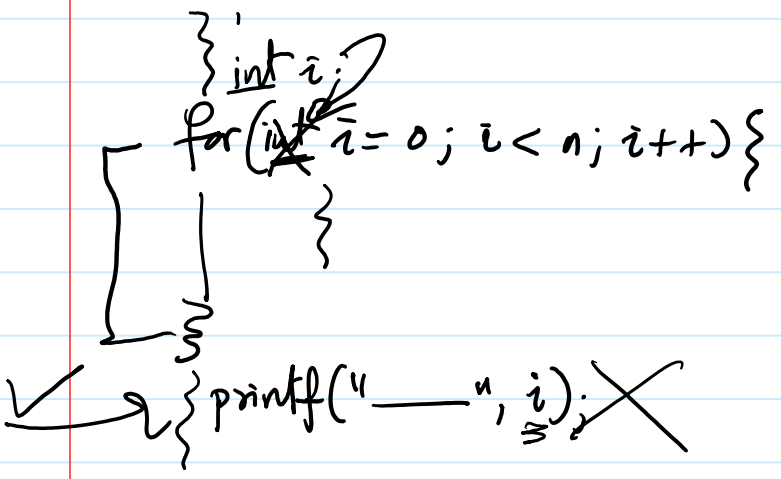
Runtime Support
less efficient

GC
Heap storage management

4 Scope (of a binding)

Sunday, September 20, 2020 6:24 PM

Q. Where in a program is a name visible/usable?




```
int z;

int gcd(int a, int b) {
    while (a != b) {
        if (a > b)
            a = a - b;
        else
            b = b - a;
    }
    return a;
} // gcd()
int v;

int main() {
    int x, y;

    // Input x and y

    int g = gcd(x, y);

    // Output g

    return 0;
} // main()
```

Scope: Region of program in which a binding is active (= accessible)

→ local variables
→ non-local variable
→ global variable

4 Static Scope: Example

Wednesday, September 23, 2020 10:16 AM

```
var N

function P1 (A1) {
  var x

    function P2 (A2) {

        function P3 (A3) {
            …
        } // P3()
        …
    } // P2()

    function P4 (A4) {

        function F1 (A5) {
            var x
            …
        } // F5()
        …
    } // P4()
    …
} // P1()
```

non-local/global

local var in P1

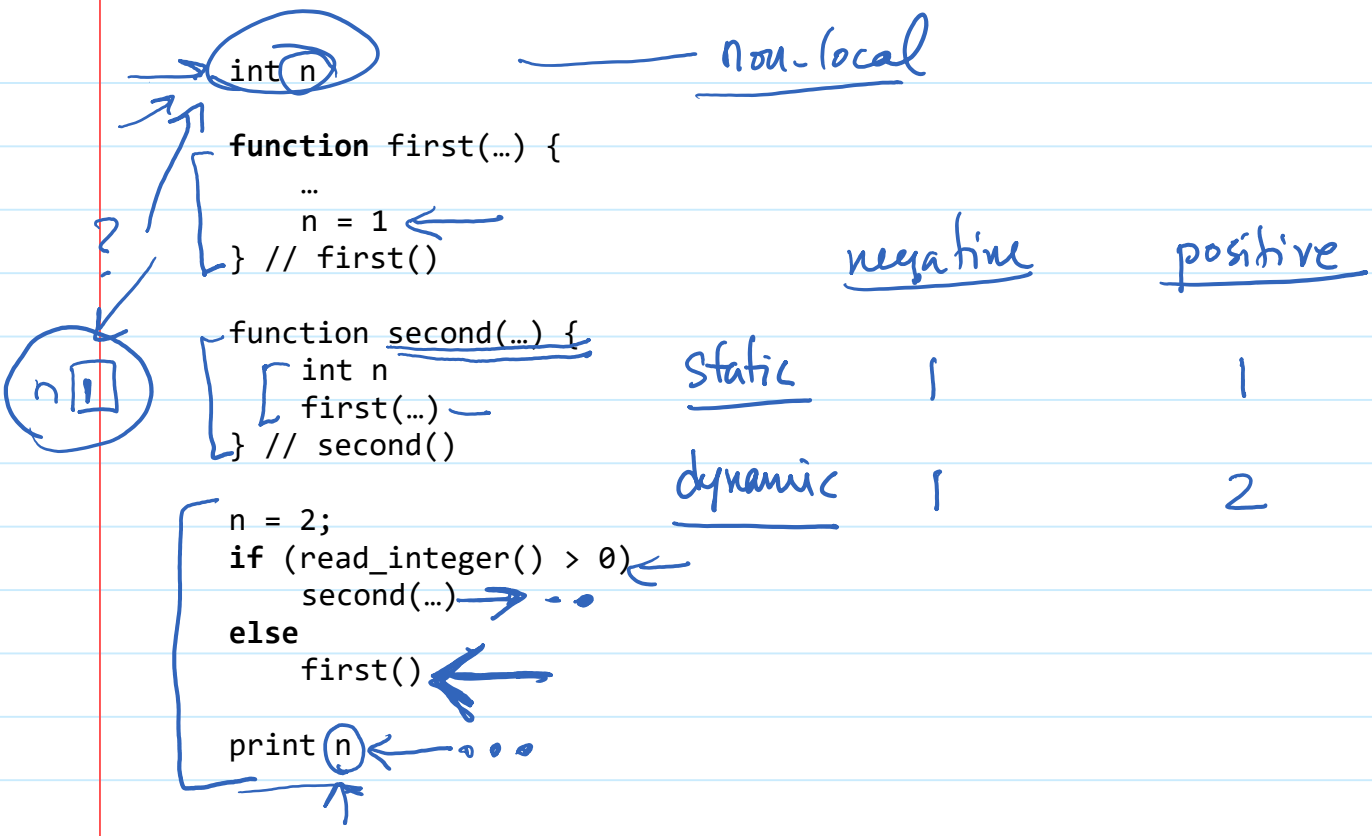
* x, P2, P4 ✓ X P2, P3, F1, x in F1

Static Scoping/Lexical Scope

When the scope of all names can be known by looking at the text of program i.e. if can be determined at compile time.

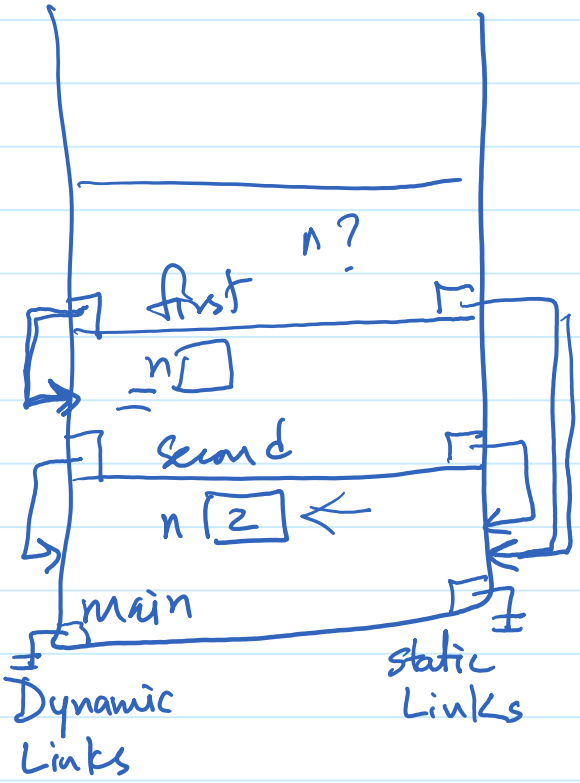
Dynamic Scoping: Example

Wednesday, September 23, 2020 10:28 AM



4 Implementing Scope Rules

Wednesday, September 23, 2020 10:40 AM




```
int n

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

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

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

print n
```