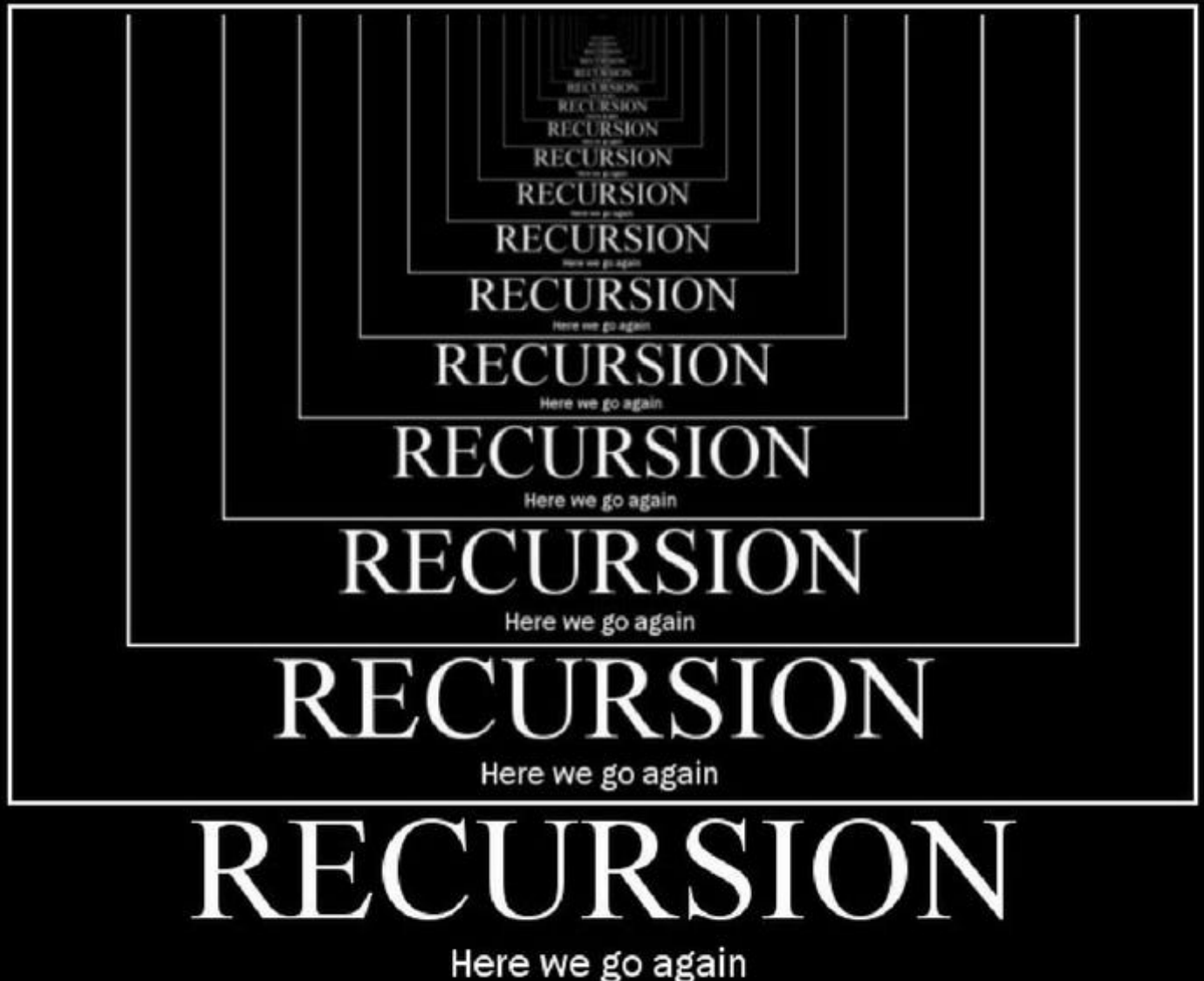
Recursion

Basics


RECURSION
Here we go again
RECURSION
Here we go again
RECURSION
Here we go again
RECURSION
Here we go again
RECURSION
Here we go again
RECURSION
Here we go again
RECURSION
Here we go again
RECURSION
Here we go again
RECURSION
Here we go again
RECURSION
Here we go again
RECURSION
Here we go again
RECURSION
Here we go again

Lecture Contents

- Definition of ***Recursion***
- What we know
  - *Recursion* in Mathematics
  - *Recursion* in Linguistics
  - *Recursion* in Java
    - `Fibonacci()`
    - `printStars()`

Recursion

- Recursion is the process of defining a problem in terms of itself.

Recursion

- the process of defining a problem in terms of itself.
- a technique or process where a function or algorithm solves a problem by calling itself with a smaller or simpler input repeatedly, until a base case or termination condition is reached.

Recursion in Mathematics

- Fibonacci numbers

$$\{0, 1, 1, 2, 3, 5, 8, 13, 21, \ldots\}$$

Recursion in Mathematics

- Fibonacci numbers

$$F_n = F_{n-1} + F_{n-2}$$

$$\{0, 1, 1, 2, 3, 5, 8, 13, 21, \ldots\}$$

Recursion in Mathematics

- Fibonacci numbers

$$F_n = F_{n-1} + F_{n-2}$$

$$F_0 = 0;\ \ F_1 = 1$$ // base case (stopping criterion)

$$\{0, 1, 1, 2, 3, 5, 8, 13, 21, \ldots\}$$

Recursion in Mathematics

- Factorial $n!$

Recursion in Mathematics

- Factorial $n! = n \times (n-1) \times (n-2) \times \ \ldots \ \times 3 \times 2 \times 1$

Recursion in Mathematics

- Factorial $n! = n \times (n-1) \times (n-2) \times \ \ldots \ \times 3 \times 2 \times 1$

$$\{1, 1, 2, 6, 24, 120, 720, 5040, 40320, \ldots\}$$

Recursion in Mathematics

- Factorial $n! = n \times (n-1) \times (n-2) \times \ \ldots \ \times 3 \times 2 \times 1$

$$F_n = n \times F_{n-1}$$

$$\{1, 1, 2, 6, 24, 120, 720, 5040, 40320, \ldots\}$$

Recursion in Mathematics

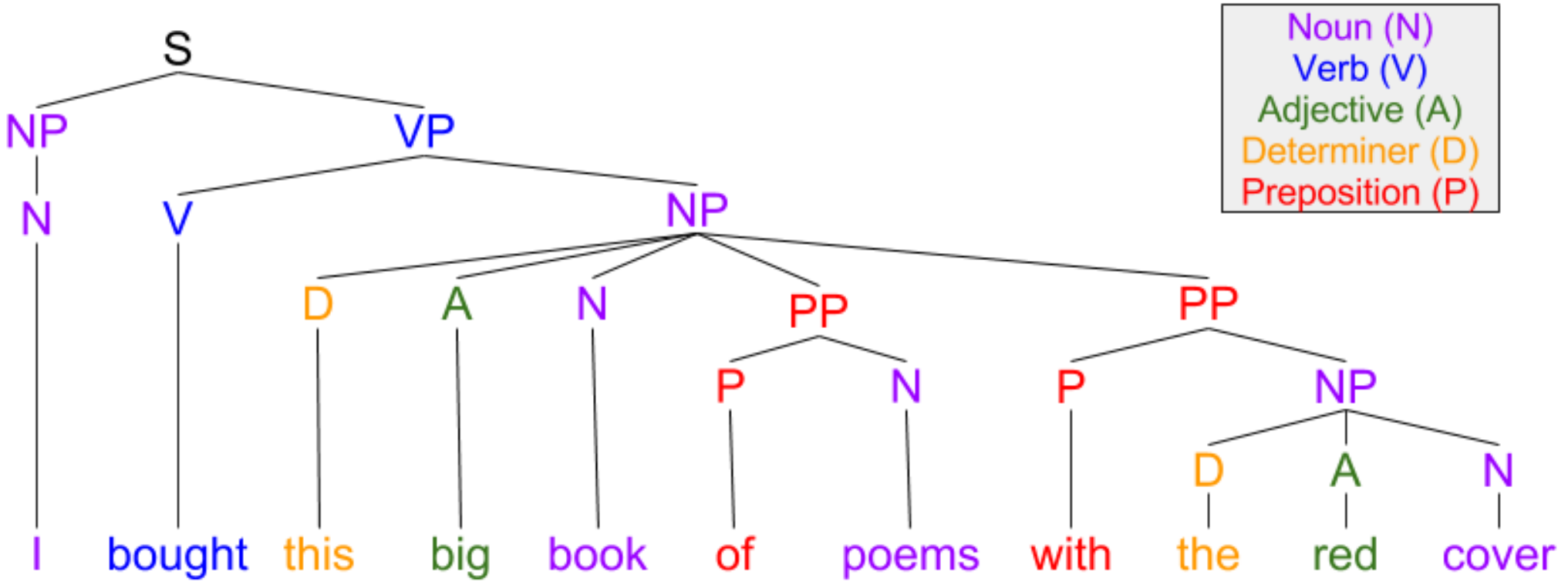
- Factorial $n!$

$$F_n = n \times F_{n-1}$$

$$F_0 = 1$$ // base case (stopping criterion)

$$\{1, 1, 2, 6, 24, 120, 720, 5040, 40320, \ldots\}$$

Recursion in Linguistics



Fibonacci in Java

$$F_n = F_{n-1} + F_{n-2}$$

$$F_0 = 0;\ \ F_1 = 1$$

Implement using → a loop in Java

```
public static int
    loopFibonacci(int n)
{




}
```

Fibonacci in Java

$$F_n = F_{n-1} + F_{n-2}$$

$$F_0 = 0;\ F_1 = 1$$

```
public static int
loopFibonacci(int n)
{
    if (n <= 1) {
        return n;
    }

    int fMinus1 = 0;
    int f = 1;

    for(int i = 2; i <= n; i++) {
        int tmp = f;
        f = f + fMinus1;
        fMinus1 = tmp;
    }

    return f;
}
```

Fibonacci in Java

$$F_n = F_{n-1} + F_{n-2}$$

$$F_0 = 0;\ F_1 = 1$$

```
public static int fibonacci(int n)
{
   if(n <= 1) {
      return n;
   } else {
      return fibonacci(n-1) +
             fibonacci(n-2);
   }
}
```

```
public static int
loopFibonacci(int n)
{
   if (n <= 0) {
      return n;
    }

   int fMinus1 = 0;
   int f = 1;

   for(int i = 2; i <= n; i++) {
      int tmp = f;
      f = f + fMinus1;
      fMinus1 = tmp;
   }

   return f;
}
```

Exercise: `printStars`

- Method `printStarts` takes an integer parameter, `n`, and prints out to the console that number of asterisks.
  - Write the method using a loop.
  - Write the method using recursion.

Recursion

Basics