```
gcd(m,n)
    limit = min(m,n)
    factorlist = []
    for each k in 1 to limit
        if k|m and k|n
            append k to
                factorlist
    report rightmost element
        in factorlist as gcd
```

```
Assume m > n
if n|m
    report n
else
    report gcd(n,
              m mod n)
```

What happens if m < n?

```
gcd(6,18)
    → gcd(18, 6 mod 18)
        = gcd(18,6)
```

Refining the naive algorithm

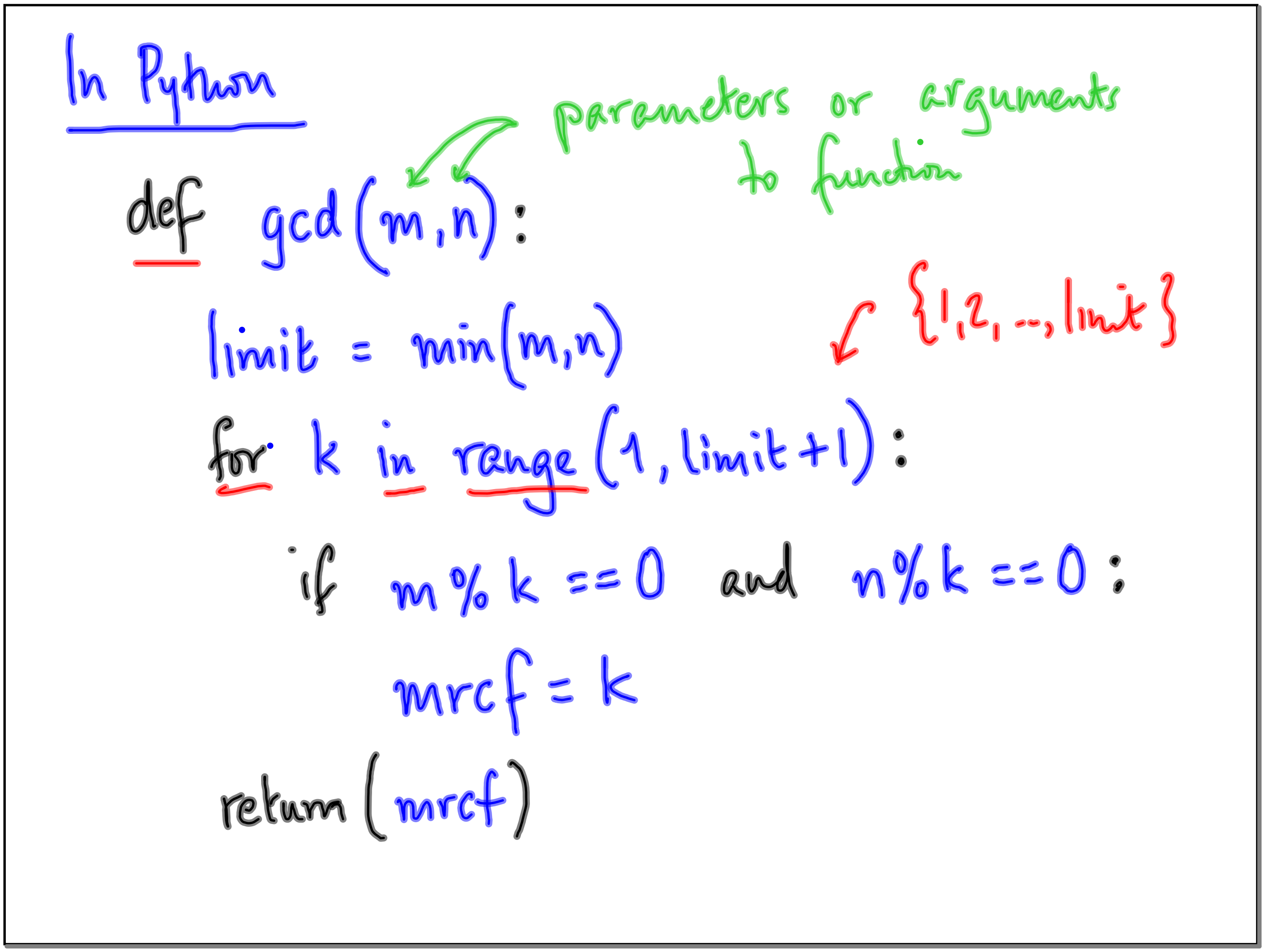
Don't need a list – sufficient to remember most recent common factor mrcf

```
limit = min(m,n)
for k in 1 to limit
    if $k|m$ and $k|n$
        set mrcf to k
report mrcf as gcd
```

Short cut:

```
for k in limit down to 1
    ...
```

Stop at first common factor



In Python

```
def gcd(m,n):
    limit = min(m,n)
    for k in range(1,limit+1):
        if m%k == 0 and n%k == 0:
            mrcf = k
    return(mrcf)
```

parameters or arguments to function

$\{1, 2, \ldots, \text{limit}\}$

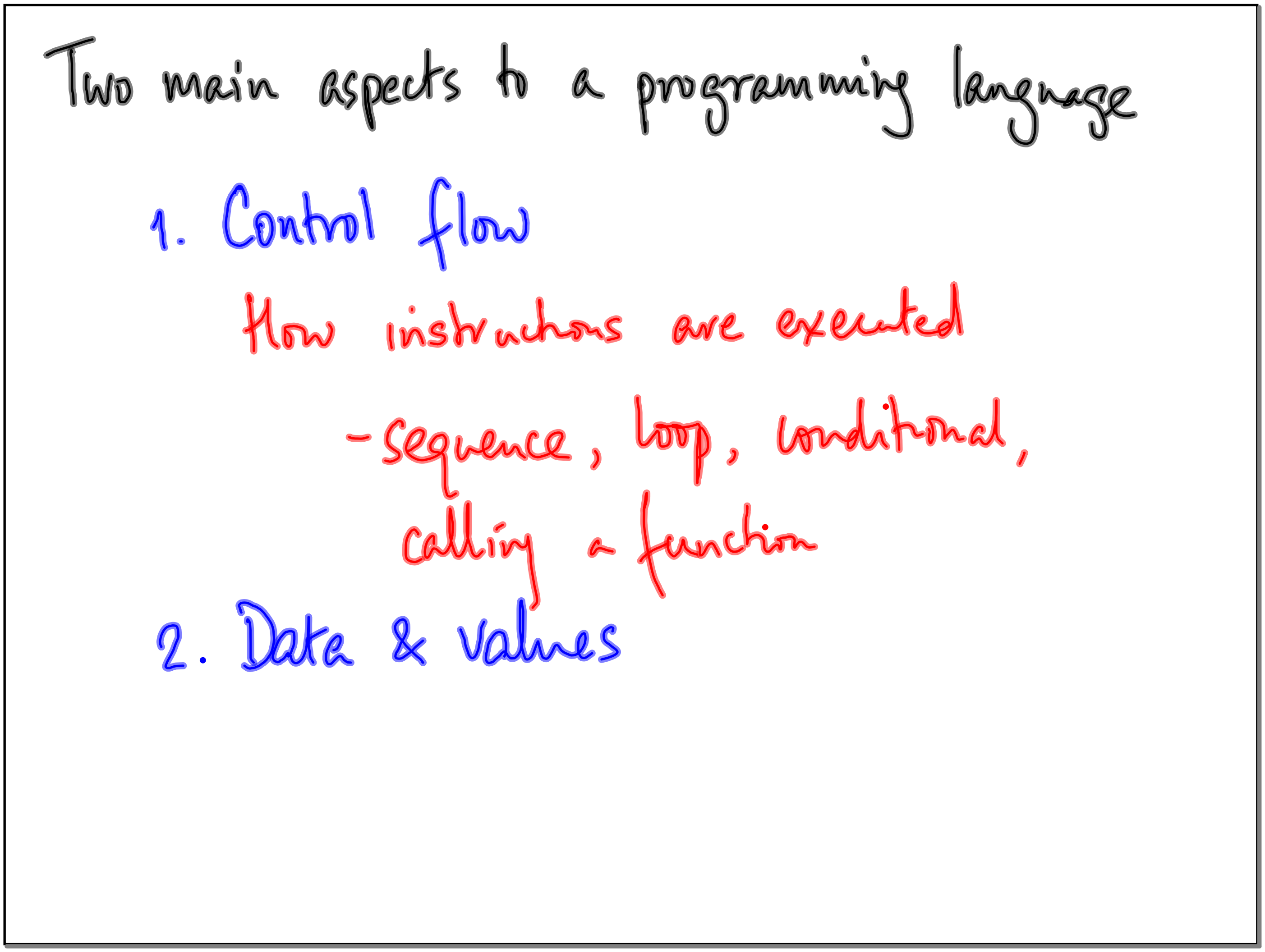
Python program

```
def function1(..):
    ≡
def function2(..):
    ≡
  ⋮
def functionM(..):
    ≡
≡ ]
```

← Anonymous function, automatically executed

```
def gcd(-)
    ≡
```

read values m & n from keyboard & report gcd



Two main aspects to a programming language

1. Control flow

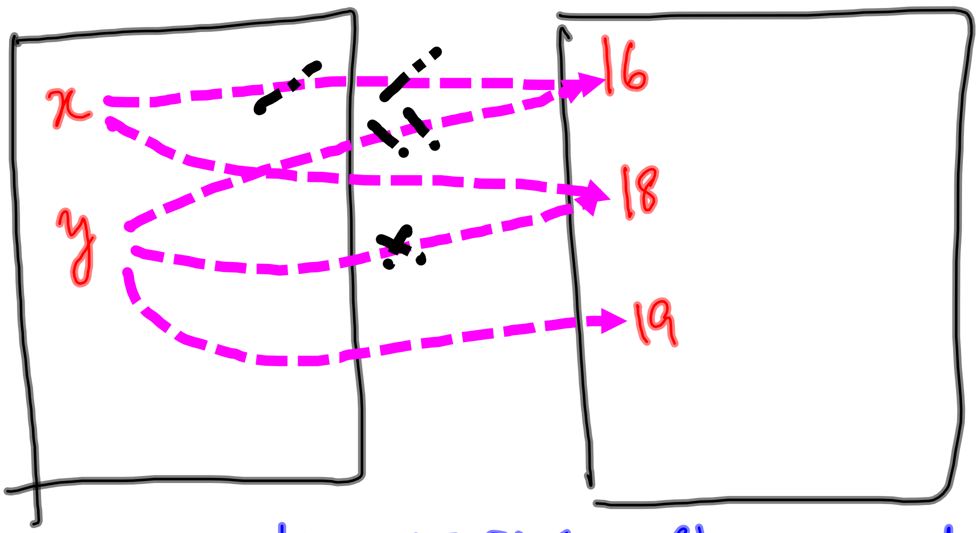
   How instructions are executed

   - sequence, loop, conditional, calling a function

2. Data & values

Data in Python.

Names Values



```
x = 16
y = x
:
x = 18
y = y+2
y = y+1
```

IMMUTABLE : Change a value ⇒ fresh copy

Simple values.

Integers Essentially unbounded $-\infty \, .. +\infty$

Real numbers

Boolean values True, False