Walk through previous lecture

Working with Images

```
from PIL import Image
image_file = Image.open("convert image.png") # open colour image
image_file = image_file.convert('1') # convert image to black and white
image_file.save('result.png')
```

PIL

Bubble Sort and Selection Sort

Fibonacci Series

Recursion

Iteration

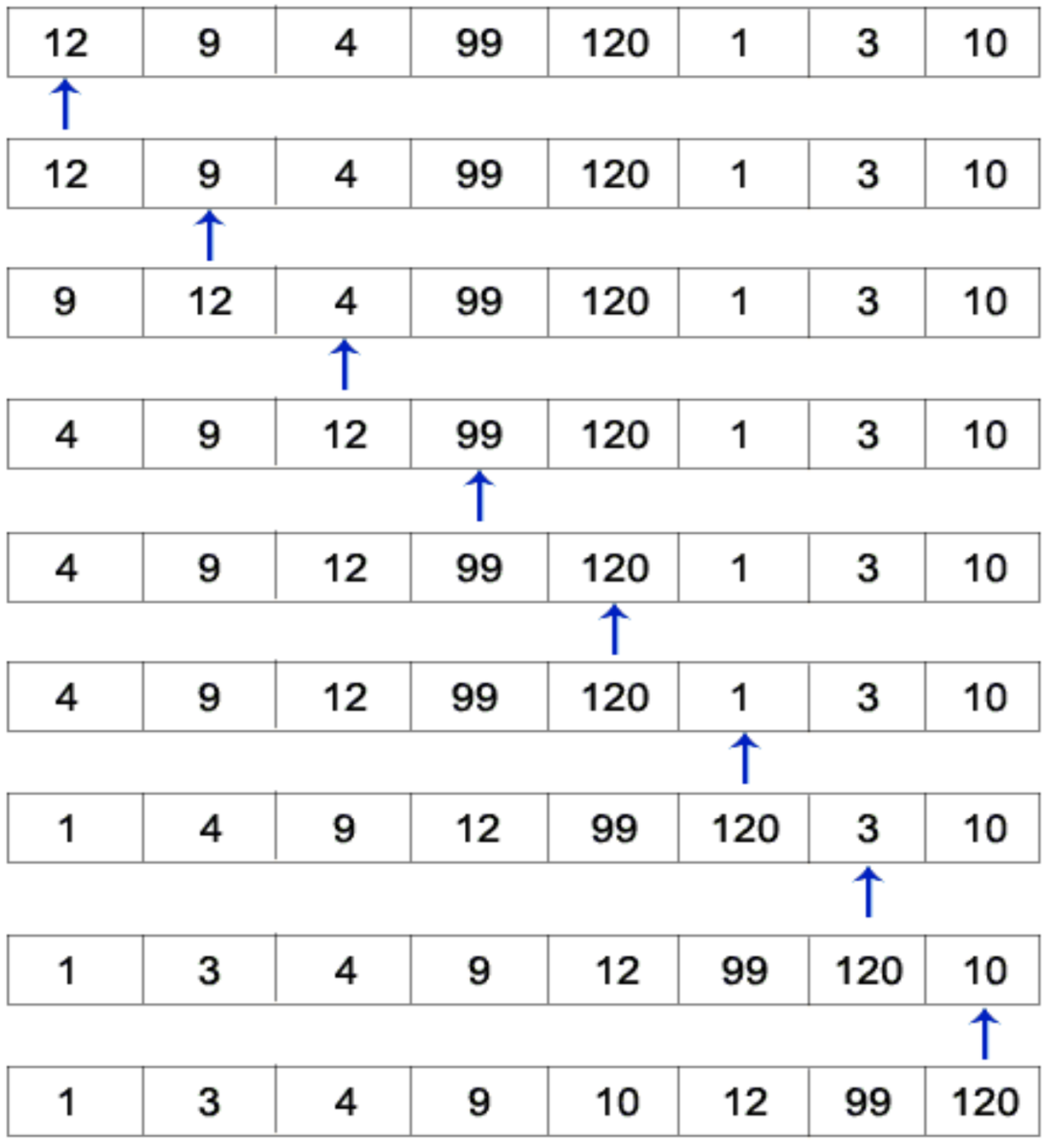
Running times

Insertion Sort

Pseudocode:

```
for i from 1 to length[A]-1 do
     value := A[i]
     j := i-1
     while j >= 0 and A[j] > value do
          A[j+1] := A[j]
          j := j-1
     done
     A[j+1] = value
done
```

Insertion Sort



Insertion Sort

```
In python:
>>> arrNumbers=[5,4,3,2,1]
... n=len(arrNumbers)
... for i in range(1,n):
...         value = arrNumbers[i]
...         j = i - 1
...         while j >= 0 and arrNumbers[j] > value:
...             arrNumbers[j + 1] = arrNumbers[j]
...             j = j -1
...         arrNumbers[j + 1] = value
... print arrNumbers
[1, 2, 3, 4, 5]
```

Introduction to python debugger

pdb

Launching pdb

Postmortem debugging:

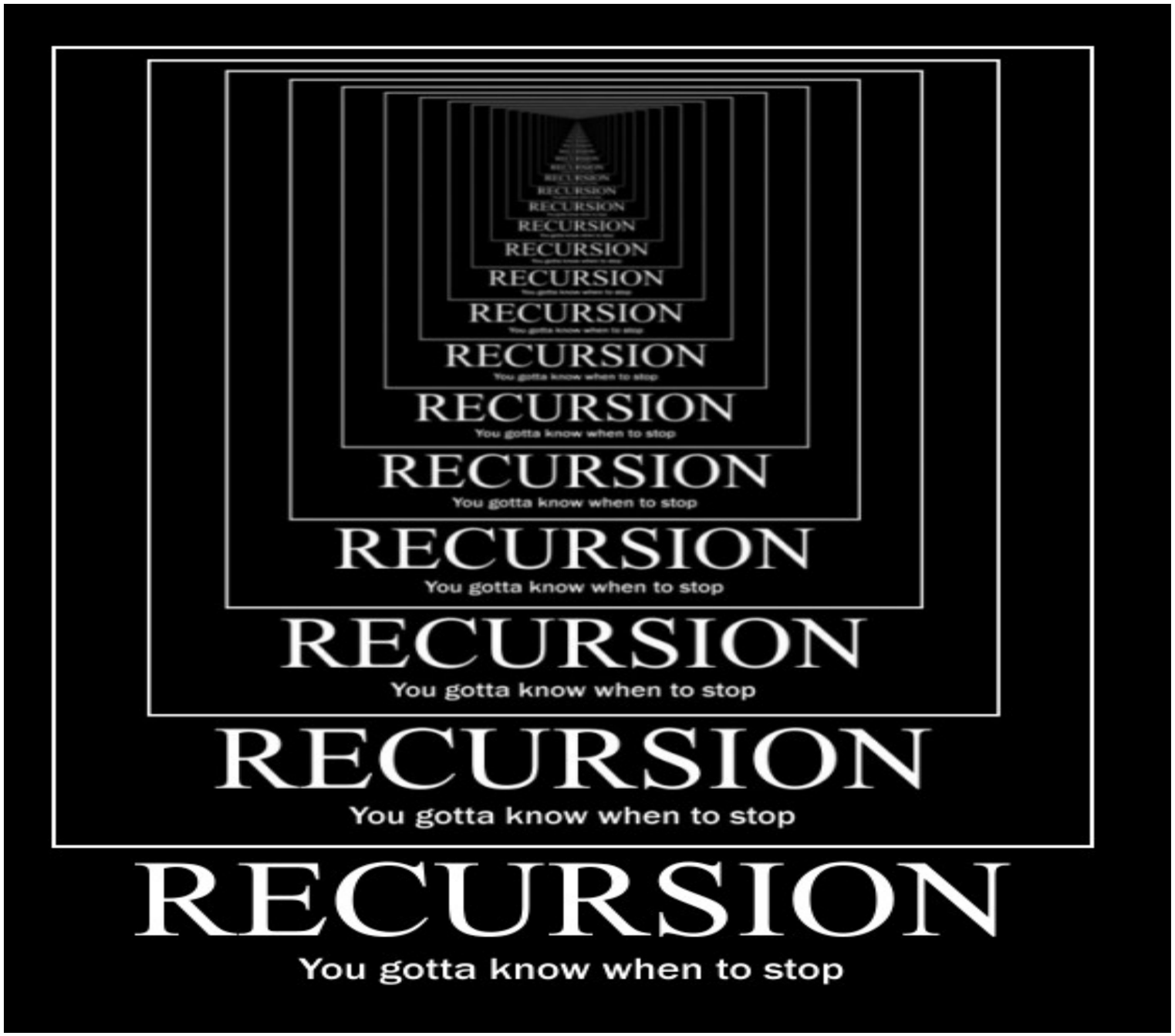
```
$ python -m pdb buggy.py
```

Or interactively:

```
>>> import buggy
>>> import pdb
>>> buggy.crash()
>>> pdb.pm()
```

Commands in pdb

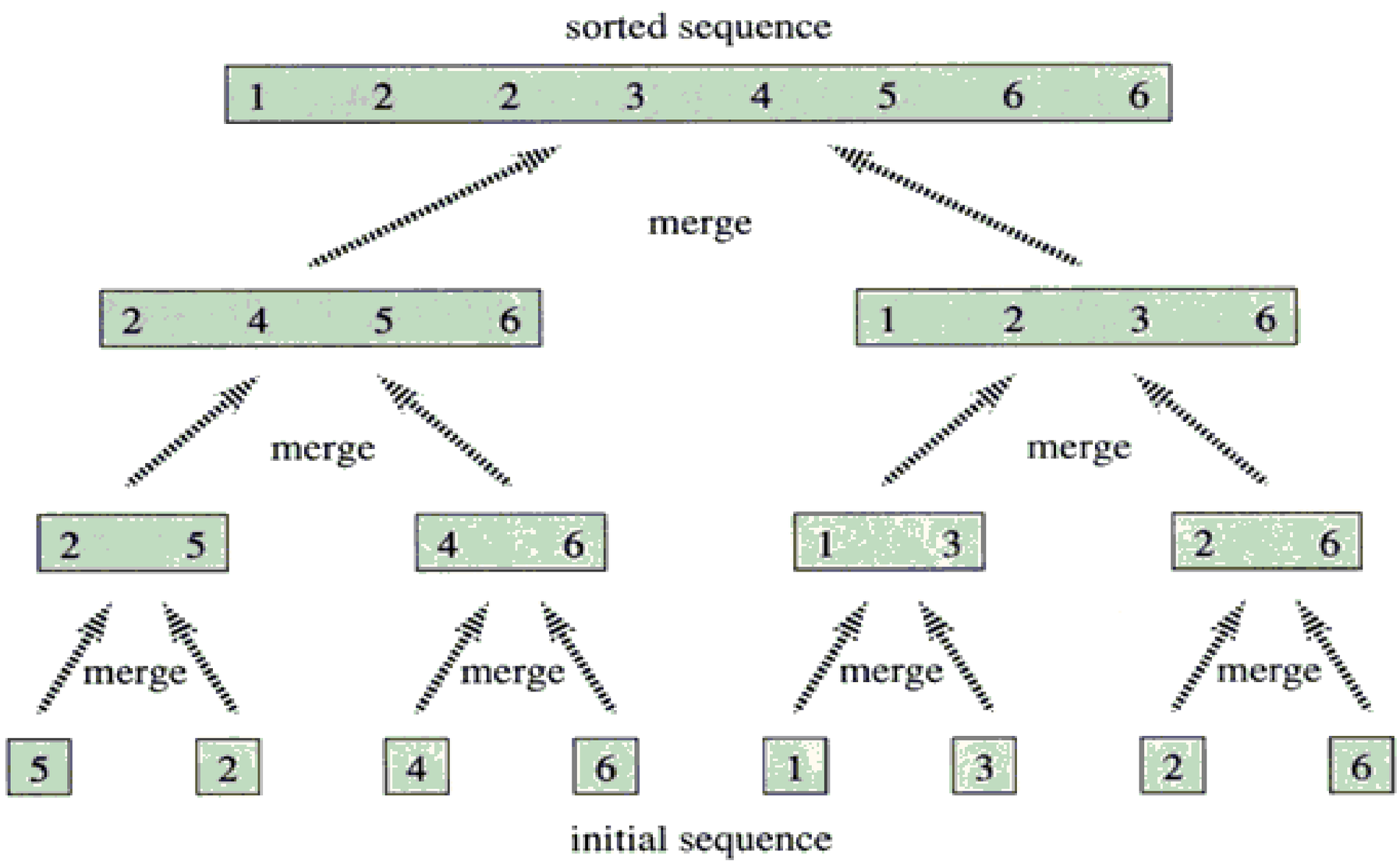
- l(ist)
- n(ext)
- c(continue)
- s(step)
- r(eturn)
- b(reak)
- q(uit)
- etc..


RECURSION
You gotta know when to stop

Merge Sort

```
On input of n elements:
     If n < 2
          Return.
     Else:
          Sort left half of elements.
          Sort right half of elements.
          Merge Sorted Halves
```

Merge Sort


sorted sequence
1 2 2 3 4 5 6 6
merge
2 4 5 6
1 2 3 6
merge
merge
2 5
4 6
1 3
2 6
merge
merge
merge
merge
5
2
4
6
1
3
2
6
initial sequence

Running Time

$$T(n) = 0, \text{ if } n > 2$$

$$T(n) = T(n/2) + T(n/2) + n, \text{ if } n > 1$$

Running Time

Example:

```
T(16) = 2 * T(8) + 16
T(8) = 2 * T(4) + 8
T(4) = 2 * T(2) + 4
T(2) = 2 * T(1) + 2
T(1) = 0
```

=64

=n (log n)

=16 (log 16)

Comparing the run times

http://www.sorting-algorithms.com/random-initial-order

to be continued...