Big O() and containers

Aahz
aahz@pythoncraft.com
http://pythoncraft.com/

Powered by PythonPoint
http://www.reportlab.org/

Before we begin...

- I'm hearing-impaired
  - Please speak slowly and clearly
- I'm also looking for a job ;-)

What is Big O()?

- Time AND Space (memory usage)
- Based on number of items to process

Why care about Big O()

- Big O() dominates your program speed and memory consumption
- Doesn't matter with hundreds or thousands of items, but does with millions

Types of Big O()

O(1) - Constant time: dict accesses, sequence lookup
O(logN) - Searching sorted lists, see `bisect` module
O(N) - Searching unsorted lists
O(NlogN) - Sorting
O(N**2), O(M*N)
O(N!), O(2**N)

General Container Info

- Container elements are references
  - Container copy does not double memory use
- Basic types: sequences and maps
  - Sequences: tuples, lists
  - Maps: dicts, sets
- Use the source, Luke
  - `Objects/` in python source tree

Lists

- list[i] is O(1)
- Amortized Constant Time
  - Over-allocation memory factor 1.5
  - Can shrink on deleting elements
- Adding/deleting list[0] is O(N)

Dicts

- Good hashing makes everything O(1)
- Bad hashing causes O(N)
- Hashing: can't use lists as keys

  ...or dicts or sets, but `frozenset()` DOES work as keys

- More over-allocation than list

  Max 2/3 full; factor is 4 to 50K items, then doubles

  Dict resizes ONLY on NEW key

Useful modules

- `collections.deque`

  O(1) at both ends

- `heapq`

  Fast for M smallest/biggest of large N

More on O(M*N)

- Code smell:

```
for i in M:
    for j in N:
        if i == j:
            ...
```

- Better:

```
tmp = set(N)  # pick the LARGER of M/N
for i in M:
    if i in tmp:
        ...
```

Some other issues

- += with strings (and other immutables)
- More bad hash