CS 252

M, 22 April 2024
Python lists

```python
a = []
for k in range(1000):
    a.append(k)
```

Does interpreter pre-allocate memory? If so, how many slots?
\[ a = [] \]

Pretend: preallocate 20 slots

For \( k \) in range(1000):
    \( a.append(k) \)

When \( k = 20 \), we ran out of slots.

What happens?

Charlie suggests:

1. Allocate a new array double-sized
2. Copy the data from old to new
3. Deallocate old

?
Number of rounds = \lceil \log_2 \frac{N}{20} \rceil

# copy ops = 20 \left( 1 + 2 + 4 + 8 + 16 + 32 \right) = 20 \left( 2^{10} - 1 \right) = 20 \left( 2^{\log_2 \frac{N}{20}} - 1 \right)
Total runtime is $O(N)$

$2^\left\lceil \log_2 \frac{N}{20} \right\rceil$ kinda like $2^{\log_2 \frac{N}{20}} = \frac{N}{20}$
What's the runtime of `a.append`? It depends.

Usually ~ $O(1)$

If you need to reallocate: $O(N)$

"Amortized constant time"
If we added fixed-size chunks instead of doubling

\[ \text{copy} \]

\[ 21 \]

\[ 40 \]

\[ 60 \]

\[ \text{copy ops} = 20 \left( 1 + 2 + 3 + \ldots + \frac{N}{20} \right) \]

\[ O(N^2) \]