3.1 analysis and big-O, lecture 2

Tuesday, September 13, 2022

Reminder: test 1 in lab next week

TODAY: nullptr

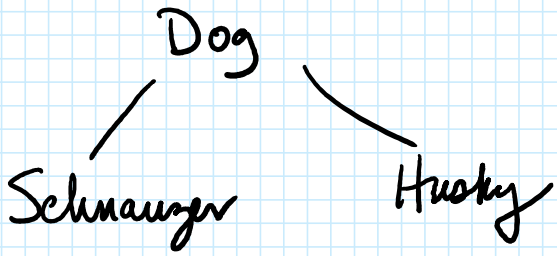
polymorphism and destructors

evaluating algorithms { empirical analysis / theoretical analysis

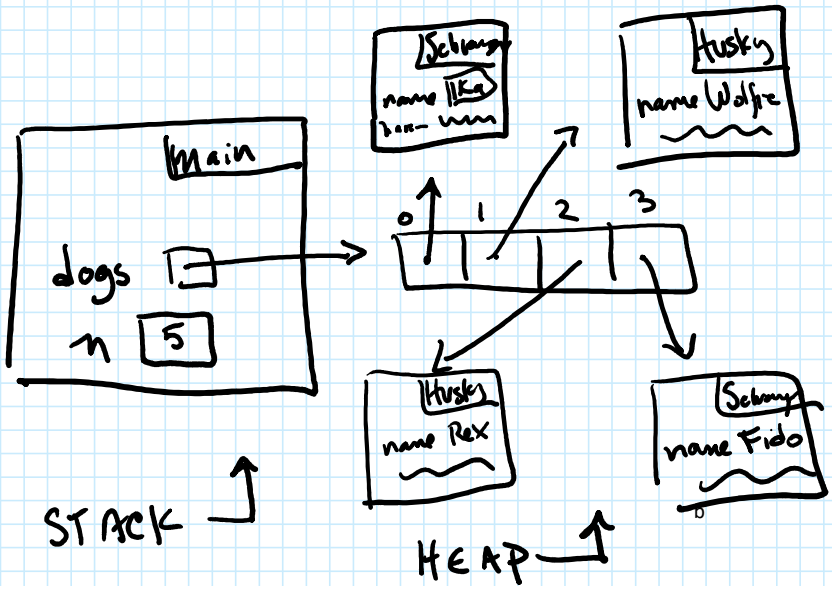
big-O definition, theoretical analysis

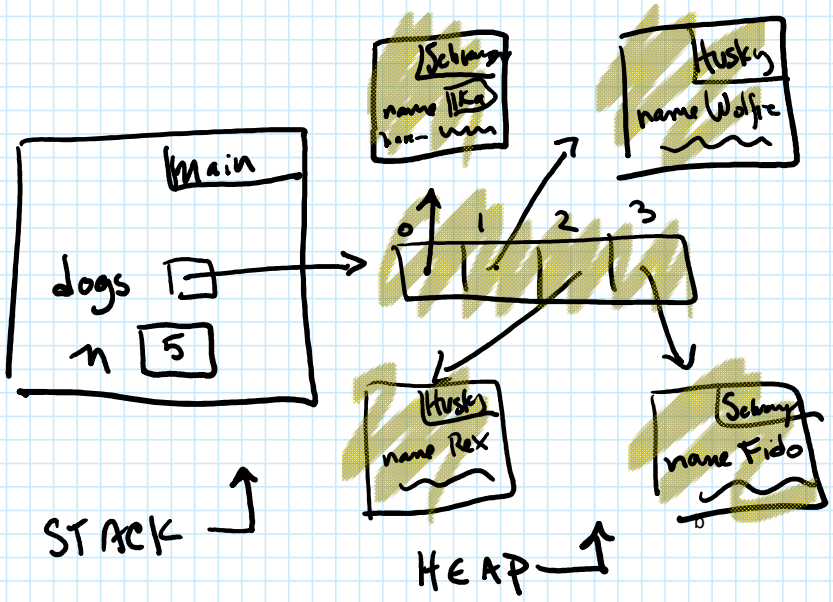
Recap: polymorphism, inheritance, destructors

With an inheritance relationship like



a pointer to the super class type Dog can point to any memory storing an object whose type is a subclass




main
dogs
n
5
0
1
2
3
name Rex
name Wolfie
name Fido
STACK
HEAP

problem: bool is_sorted(int* array, int size)

returns { true, if values are in ascending order
false, if values are NOT in ascending order

ex

```
int a[5] = {2, 4, 6, 8, 10};
int b[3] = {13, 1, 7};
```

is_sorted(a,5) true!

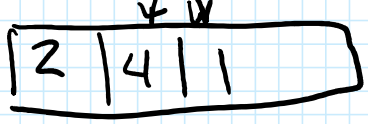
is_sorted(b,3) false!

How can we choose which version of an algorithm is "best"?

- shortest time to run
- memory usage
- readability
- correctness ALWAYS!

version 1: correct ✓

version 2: correct ✓



version 3: correct ✓

Empirical Analysis

Want to compare versions to find which is most efficient.

- let's run some experiments and time them
- vary the size of the array

version \ arraysize	10	100	1000	10,000	50k
1	0.005	0.005	0.009	0.13	2.9
2	0.004	0.004	0.004	0.004	0.005 ??!
3	0.008	0.016	0.093	0.859	4.2

version 2 is the fastest

1 & 3 are getting slower more,

3 is slowing a LOT as inputsize increases

notes on empirical analysis

- clock time is not the clearest measure

hard to control conditions for equal runs

- can only run on a limited set of values
- must implement each algorithm to test it

THEORETICAL ANALYSIS

- abstract away from a particular language
- analyze how much work the algorithm requires

work = # of steps

- count the # of steps in an algorithm
- focus on key steps

  is_sorted: key step is comparison between elements
- let n represent the size of the problem

Version 1

i loop goes from i=1 to i= size - 1

  j loop goes from j=i+1 to j=size-1

    comparison

How many comparisons are done in total?

Version 2:

i loop goes from i=0 to size-1

comparison

How many comparisons are done in total?

version 3

i loop goes from i=0 to size-1

10K comparisons

How many comparisons are done in total?