CPSC 212 – SPRING 2012

Data Structures & Alg's

Alg. Analysis

Theory of Comp.

LINKED LISTS

HASH TABLES

TREES

(recurrence relations)

(AVL, kd-tree)

GRAPHS, HEAPS

Algorithms:

- sorting, heapsort
- searching
  (find min, max,
  find k. nearest)

Alg. Design

→ design efficient alg.

↓

min. (perhaps optimal)

Use of resources

((PU, RAM, disk)

↓

Speed

e.g. linear search:

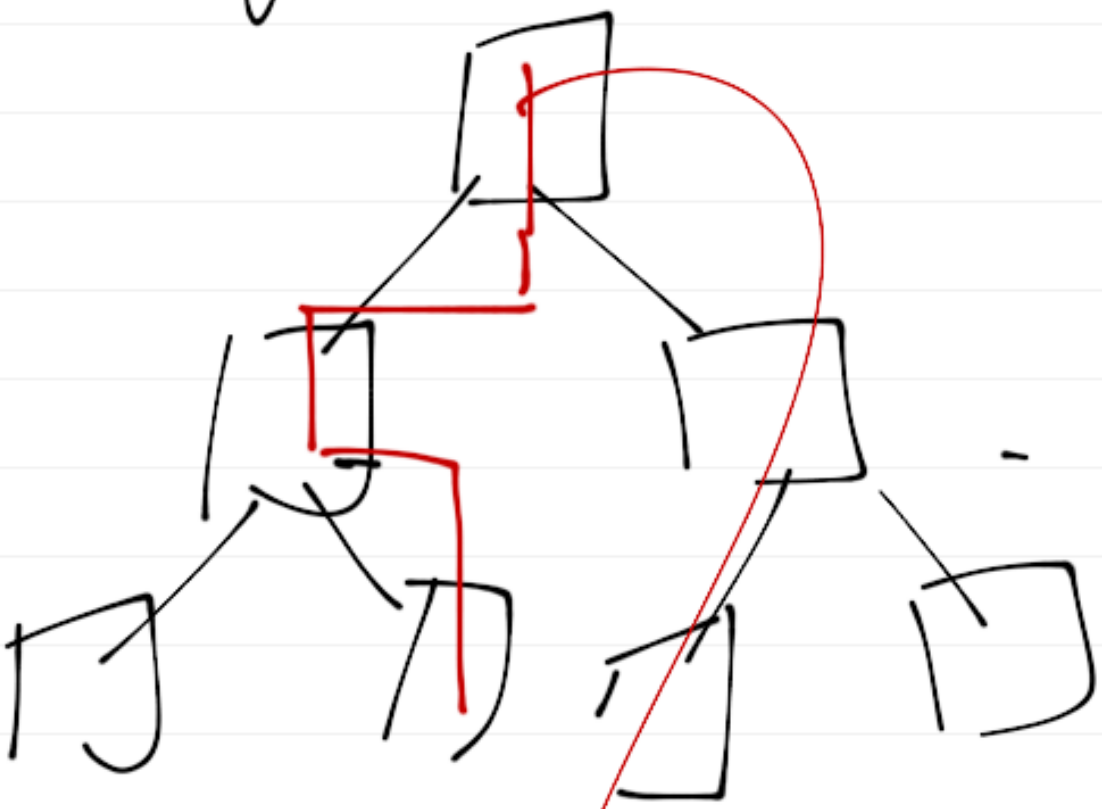
```
for(i=0; i<arr.size(); i++)
    if(arr[i]==target)
        stop;
```

n

O(n)

↑ take n steps in worst case

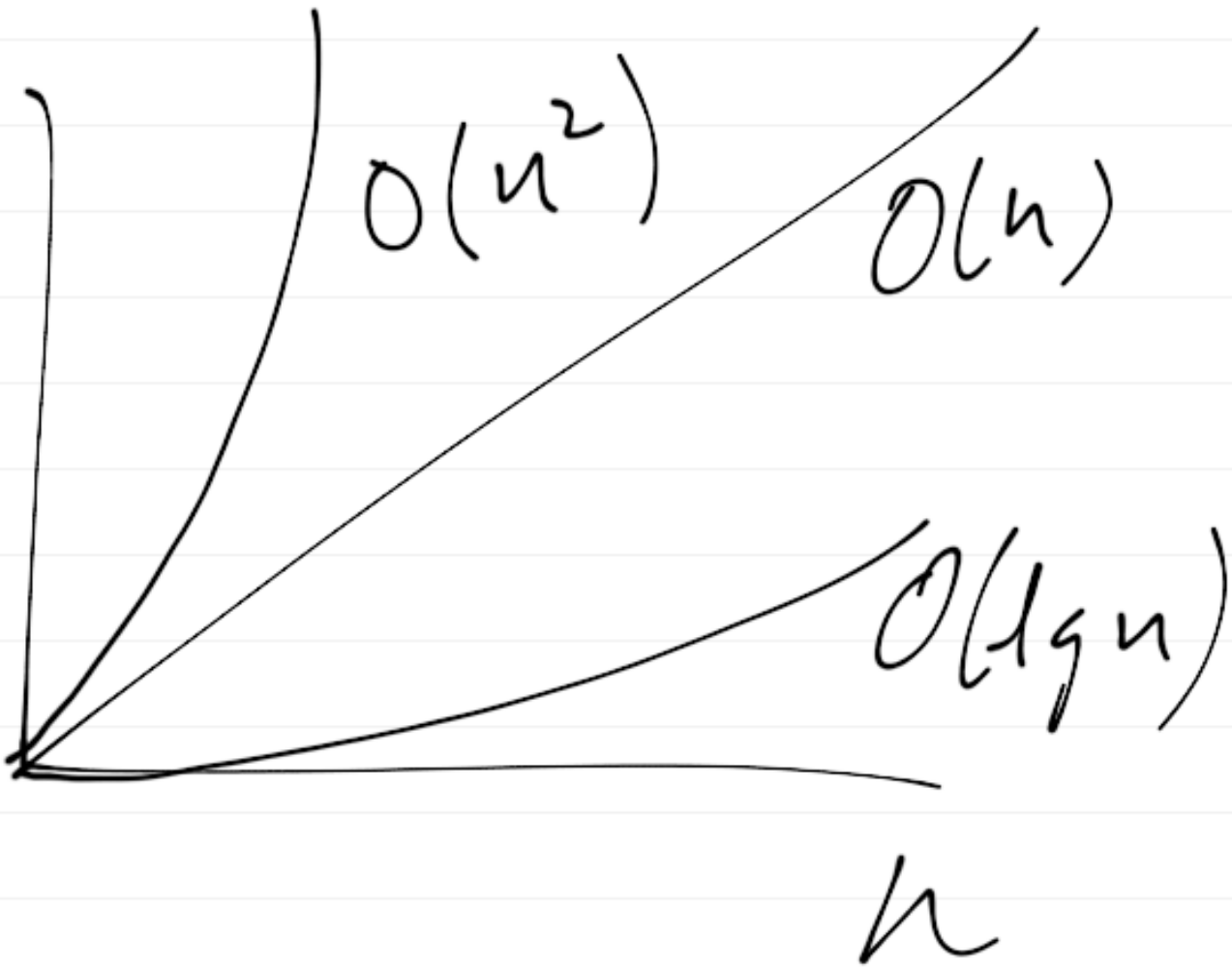
- if using tree



at each step,

excluding $\frac{1}{2}$ n

$\Rightarrow O(\lg n)$


O(n²)
O(n)
O(lg n)
n

Intro to C++:

timer_t class

↳ look up in text:

"the big three"

for any object (class)

first thing: implement

the big three

- big 3:

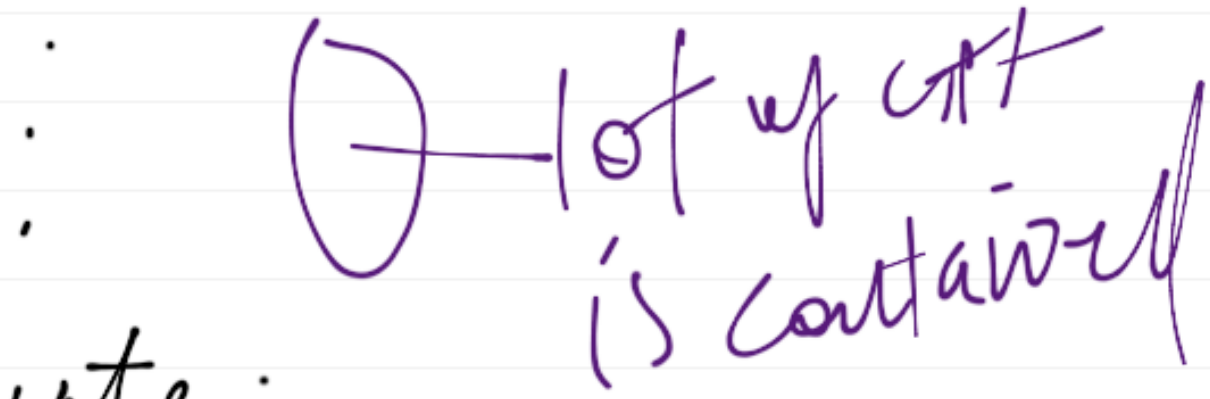
destructor,

copy constructor,

operator=

assignment op.

```
class timer_t {
  :
private:
  double ts; // start time
  double te; // end time
  double tf; // total time
};
```



similar to typedef struct

```
class timer_t {
public:
  timer_t(double s=0.0,
          double e=0.0,
          double t=0.0) : \
    ts(s), \
    te(e), \
    tt(t) { };
```

constructor

```
typedef struct {
    double ts, te, tt;
} timer_t;
```

vs.

```
class timer_t {
    private:
        double ts, te, tt;
    public:
        timer_t () { };
        void start ();
        void end ();
        :
};
```

public member functions

Usage: (C++)

```
timer_t timer;

timer.start();
timer.end();
```

VS

```
timer_t timer;

timer_start(&timer);
timer_end(&timer);
```