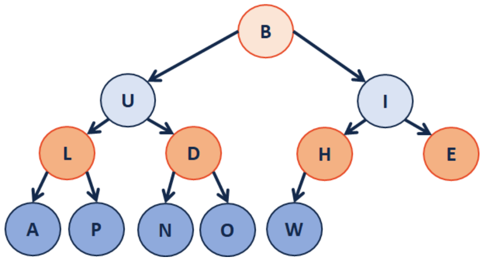
#29: Disjoint Sets Intro

April 2, 2018 · *Wade Fagen-Ulmschneider*

Building a Heap with an Array of Data

- Assumption: Data already exists as an unsorted array in memory.
- Goal: Organize the data as a minHeap as fast as possible.



-	B	U	I	L	D	H	E	A	P	N	O	W		

Solutions:

1. Sort the array, O(n lg(n))
2. Use Heap::insert for every element, O(n lg(n))
3. Use a heapifyDown strategy on half the array:

```
Heap.cpp (partial)
template <class T>
void Heap<T>::buildHeap() {
  for (unsigned i = _parent(size_); i > 0; i--) {
    heapifyDown(i);
  }
}
```

Theorem: The running time of buildHeap on array of size n is:

____________.

Strategy:

Define S(h):
Let **S(h)** denote the sum of the heights of all nodes in a complete tree of height **h**.

S(0) =

S(1) =

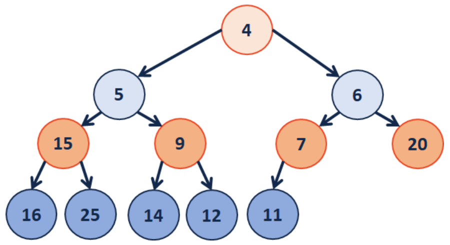
S(2) =

S(h) =

Proof of S(h) by Induction:

Finally, finding the running time:

Heap Sort



Algorithm:

1.

2.

3.

Running time?

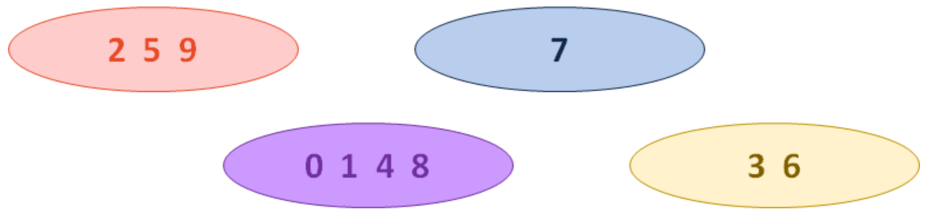
Why do we care about another sort?

Disjoint Sets

Let **R** be an equivalence relation on *us* where **(s, t) ∈ R** if **s** and **t** have the same favorite among:

{ ____, ____, _____, ____, _____, ____ }

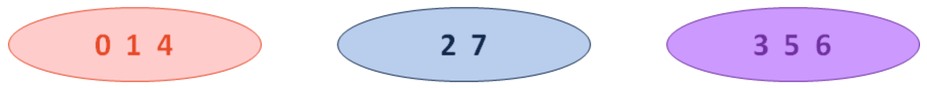
Examples:



Building Disjoint Sets:

- Maintain a collection S = {s_0, s_1, ... s_k}
- Each set has a representative member.
- ADT:

```
    void makeSet(const T & t);
    void union(const T & k1, const T & k2);
    T & find(const T & k);
```



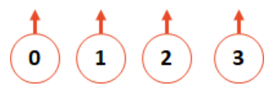
[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]

Operation: find(k)

Operation: union(k1, k2)

Implementation #2:

- Continue to use an array where the index is the key
- The value of the array is:
  - **-1**, if we have found the representative element
  - **The index of the parent**, if we haven't found the rep. element



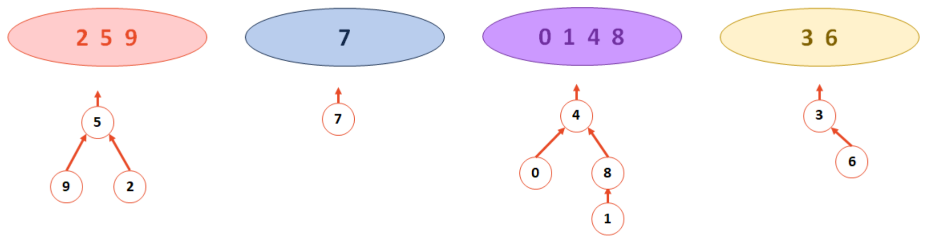
[0]	[1]	[2]	[3]

[0]	[1]	[2]	[3]

[0]	[1]	[2]	[3]

[0]	[1]	[2]	[3]

Example:



4	8	5	6	-1	-1	-1	-1	4	5
[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]

...where is the error in this table?

CS 225 – Things To Be Doing:

1. MP5 deadline tonight Monday, April 2nd
2. Theory Exam 3 starts tomorrow (Tuesday, April 3rd)
3. lab_heap starts on Wednesday
4. Daily POTDs are ongoing!