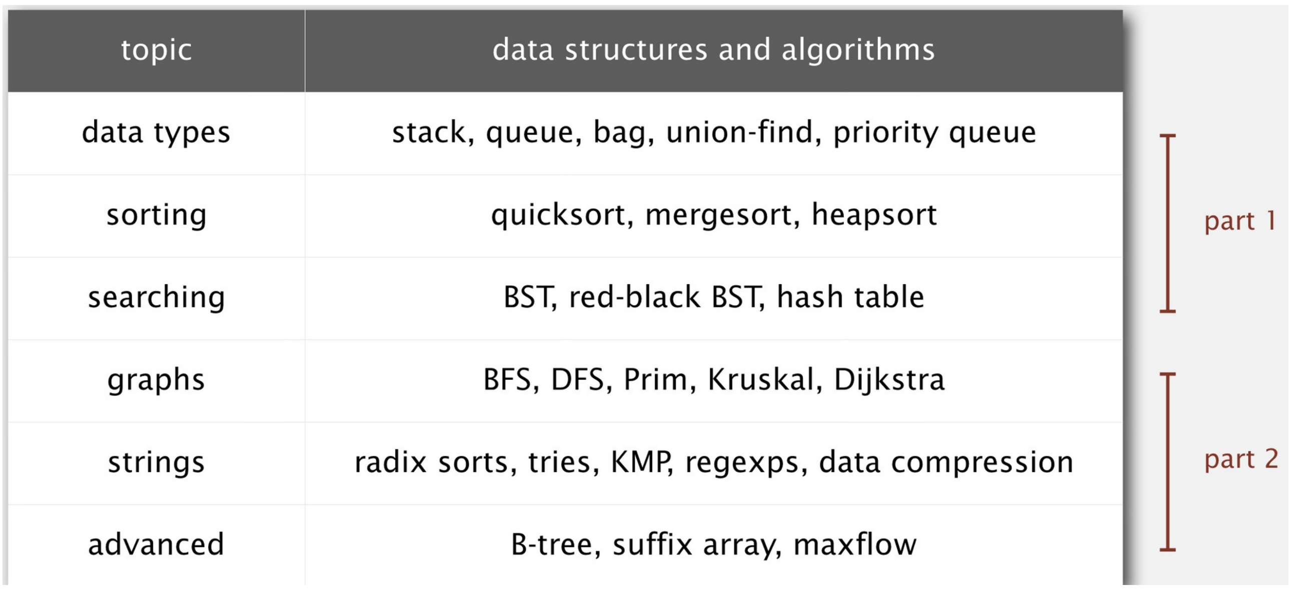
Data Structure and Algorithm

Algorithm: Method of solving a problem
Data Structure: Method to store information



topic	data structures and algorithms
data types	stack, queue, bag, union-find, priority queue
sorting	quicksort, mergesort, heapsort
searching	BST, red-black BST, hash table
graphs	BFS, DFS, Prim, Kruskal, Dijkstra
strings	radix sorts, tries, KMP, regexps, data compression
advanced	B-tree, suffix array, maxflow

part 1
part 2

Dynamic Connectivity

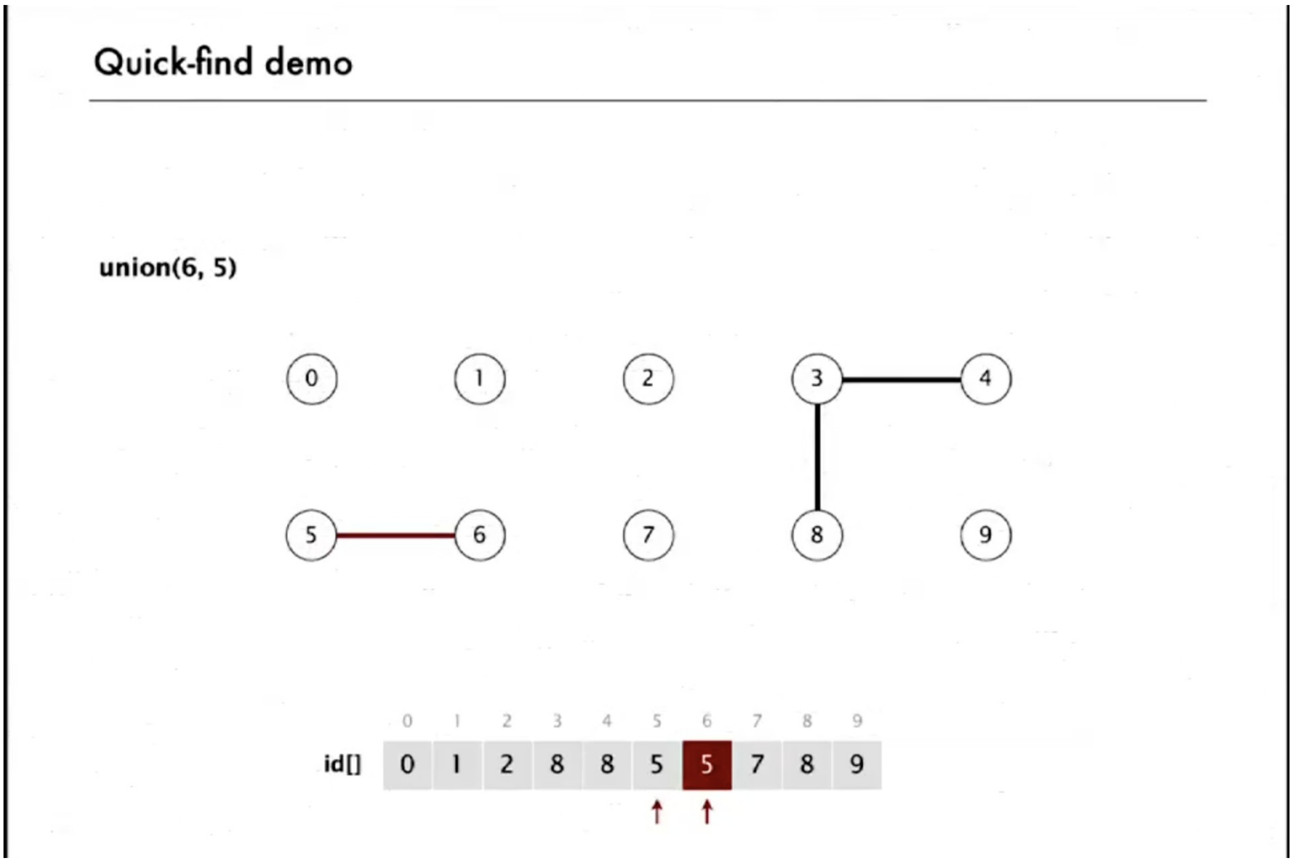
An algorithm that defines if there is a path between object
Connected component is a maximal set of objects mutually connected

- union command will connect two smaller component to form a much larger component example is union(2,5) will connect 2 to 5

Snippet example of union find

public class UF
void union(int p, int q) //A method to connect p to q
boolean connected(int p, int q)// will return a boolean depending if p and q are connected or not

In an array, the second one will be change to the first on



For union(2,1) the value of 1 will replace the value of 2, the value of 1 is 1, so the value of 2 will be 1.

connected(5,6) will return true because the values of the index 5 and 6 is 5.
connected(3,4) will be true, connected(8,9) will be false

union(5,0) the value of the second will replace the value of the first, in this case, the value of 0 which is 0 will replace the value of 5, therefore the value of 5 will be 0, also 5 is connected to 6, therefore the value of 6 will also be 0

Code

```
public class QuickFindUF{

//Define a global variable
private int[] Id;

public QuickFindUF(int N) {
//Create a new ArrayList
Id = new int[N];
for (int i = 0; i < N; i++){
/*
The loop will start counting from 0 to the number set. For Example 5;
Id[0] = 0;
Id[1] = 1;
Id[2] = 2;
Id[3] = 3;
Id[4] = 4;
*/
Id[i] = i;
}
}
//A boolean which will check if the numbers are equal
public boolean connected(int p, int q){
return Id[p] == Id[q];
}

/*
union(5,6)
First let treat 5 and 6 as keys in the array
int pid = Id[5] will get the value of pid and let assume the value is 7
int qid = Id[6] will get the value of qid and let assume the value is 8

for (int i = 0; i <10; i++)

if the value of Id[i] is equal to pid that is 7,
then the value of Id[i] will be updated with qid

So we are replacing the value of the pid with the value of the qid

*/

public void union(int p, int q){
int pid = Id[p];
int qid = Id[q];
for (int i = 0; i <Id.length; i++){
if (Id[i] == pid){
Id[i] = qid;
}

}
}





}
```

Drawbacks
QuickFind is slow, Quadratic time don't scale with time and technology