CSE 332
Summer 2018
Final Review

1 Parallel Code

Explain the steps you would use to perform the following tasks in parallel. Your algorithm should have the best possible $O()$ span, but you need not worry about constant factors in this problem. You may assume you have access to already allocated auxiliary arrays as needed, and may alter the input array. Use the following parallel code patterns discussed in class:

- `out = map(f, arr)` Applies `f` to every element of `arr`, storing the results in `out`.
- `out = reduce(baseFn, combineFn, arr)` Given `baseFn` on a single element and `combineFn` on two arrays (one of which being `arr`, the other the output of `baseFn`), `reduce` stores the results in `out`.
- `out = parallelPrefixSum(arr)` Runs ParallelPrefixSum on `arr`, storing the results in `out`.
- `out = pack(condition,arr)` Given `condition`, performs a pack/filter on `arr`, storing the results in `out`.

1. **Input**: An array of integers, `arr`
   Output:: An array where index i is the sum of the prime numbers in `arr` at indices $0, ..., i$.

2. **Input**: `arr`, an array of integers
   Output: True if `arr` is a valid (0-indexed) representation of a binary min-heap.

3. **Input**: an array `arr` **Output**: two arrays, one with elements less than k the other with elements greater than k.

4. **Input:** `arr` an array of integers
   Output: an array containing **peak** elements of `arr`. An element i is a peak element if `arr[i-1]` $<$ `arr[i]` and `arr[i+1]` $<$ `arr[i]`.

2 Graph problems

1. You and your trusty Dragonite have just finished your training outside the Pokémon League. You now feel prepared to take down the Elite Four. There's just one problem – you've earned none of your badges.

   Your goal is to visit each of the n cities with gyms, crush all the gym leaders, and return to the Pokémon League as quickly as possible.

   Describe a graph representation of this problem (For example, what are the vertices and edges? Is the graph weighted?).

   Can you design an algorithm to quickly determine the best possible route? If so describe it (you may use any algorithm discussed in class as a black box). If not, informally justify why such an algorithm isn't likely.

2. Having just finished summer quarter, you and your friends celebrate with a trip to Disney Land. You decide to have a contest. Your goal is: starting from the entrance

   (a) Ride at least two distinct rides

   (b) Make it to Splash Mountain

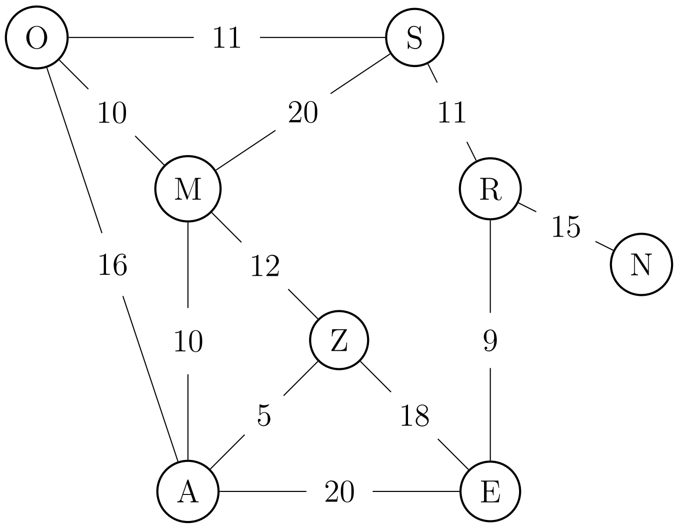
   Whoever arrives first is the winner. You have an encyclopedic knowledge of Disney Land, thus you know how long it takes to walk from any ride to any other, and moreover you know how much time it takes to go on any ride.

   Describe a way to represent this problem as a graph. Then either describe an algorithm to run to solve the problem, or informally justify that such an algorithm isn't likely.

3 Running Graph Algorithms

3.1 MST

Consider the following graph:

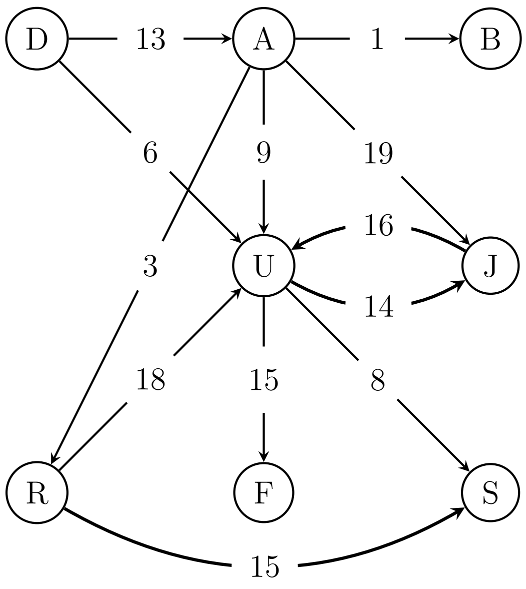


Find an MST of this graph using both of the two algorithms we've discussed in lecture. Make sure you say which algorithm you're using and show your work.

Continue your work here:

3.2 Dijkstra

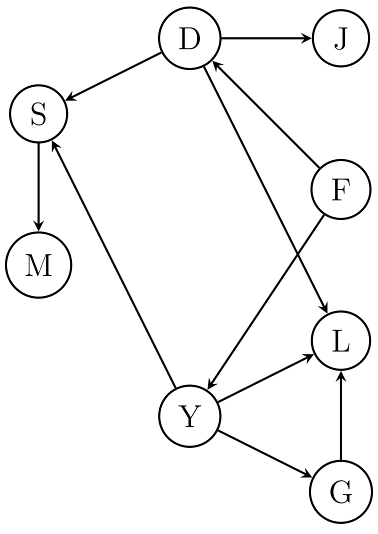
Consider the following graph:



Use Dijkstra's Algorithm to find the lengths of the shortest paths from D to each of the other vertices. For full credit, you must show your work at every step. Break ties alphabetically.

3.3 Topo Sort

Consider the following graph:



Find a topological sort of this graph.

4 Hash table

1. Suppose we have a hash table that uses separate chaining and has an internal capacity of 10 (do NOT worry about resizing for this problem). Assume that each bucket is a linked list where new elements are added to the front of the list.

   Insert the following elements in the EXACT order given using the hash function

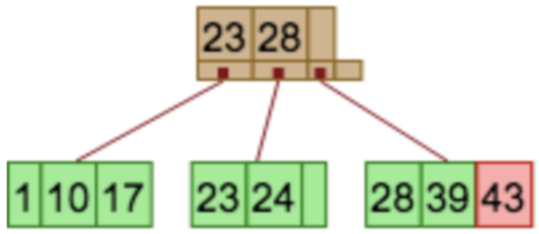
   $h(x) = x$:

   $$98, 18, 68, 21, 38, 8, 9, 11$$

2. Repeat the same insertions with quadratic probing.

5 B-tree insert and math

Given the following existing B-tree (values not shown):



1. Insert the following keys into the B-tree, showing work:
   $3, 37, 45, 7, 40, 31$

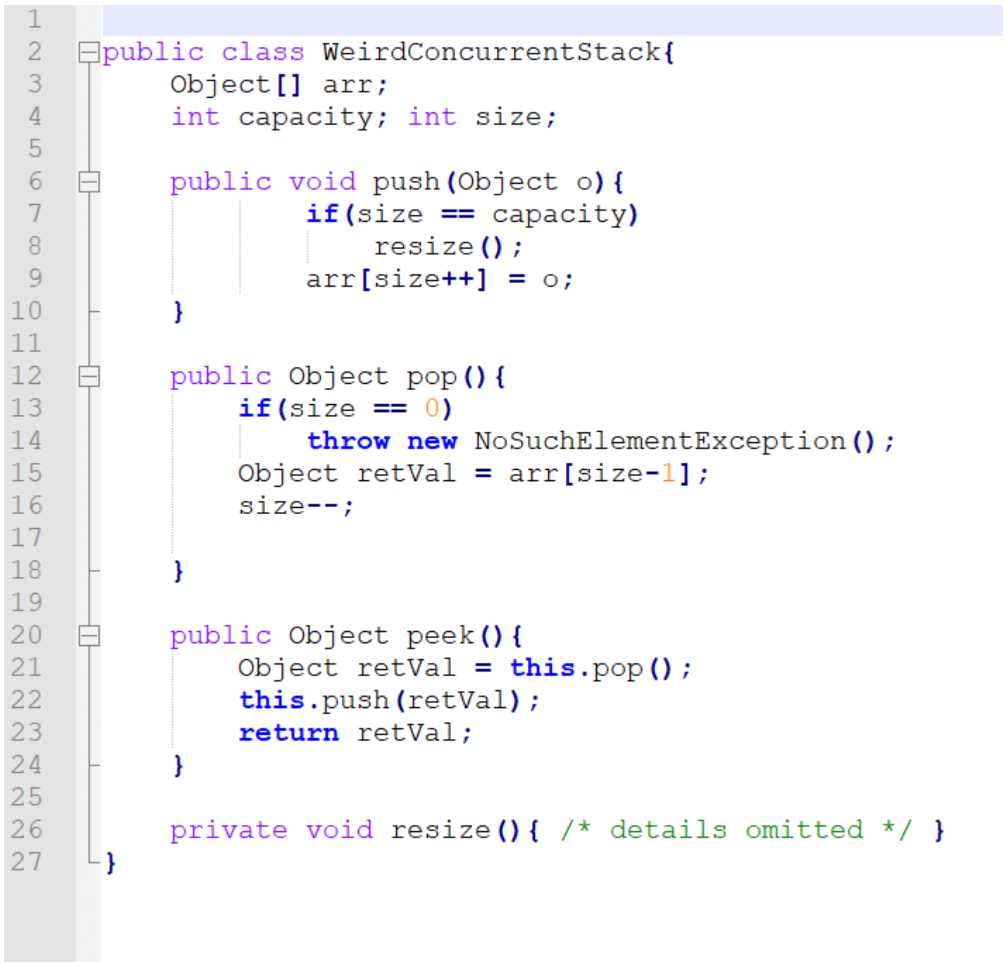
2. Delete the following keys from *the initial* B-tree:
   $10, 28, 24, 1, 39, 43$

3. Given the following parameters for a B-Tree with $M = 16$ and $L = 13$:
   - Key Size = 4 bytes
   - Pointer Size = 2 bytes
   - Data Size = 12 bytes per record (includes the key)

   Assuming that M and L were chosen appropriately, what is the likely page size on the machine where this implementation will be deployed? Give a numeric answer based on two equations using the parameter values above.

6 Concurrency

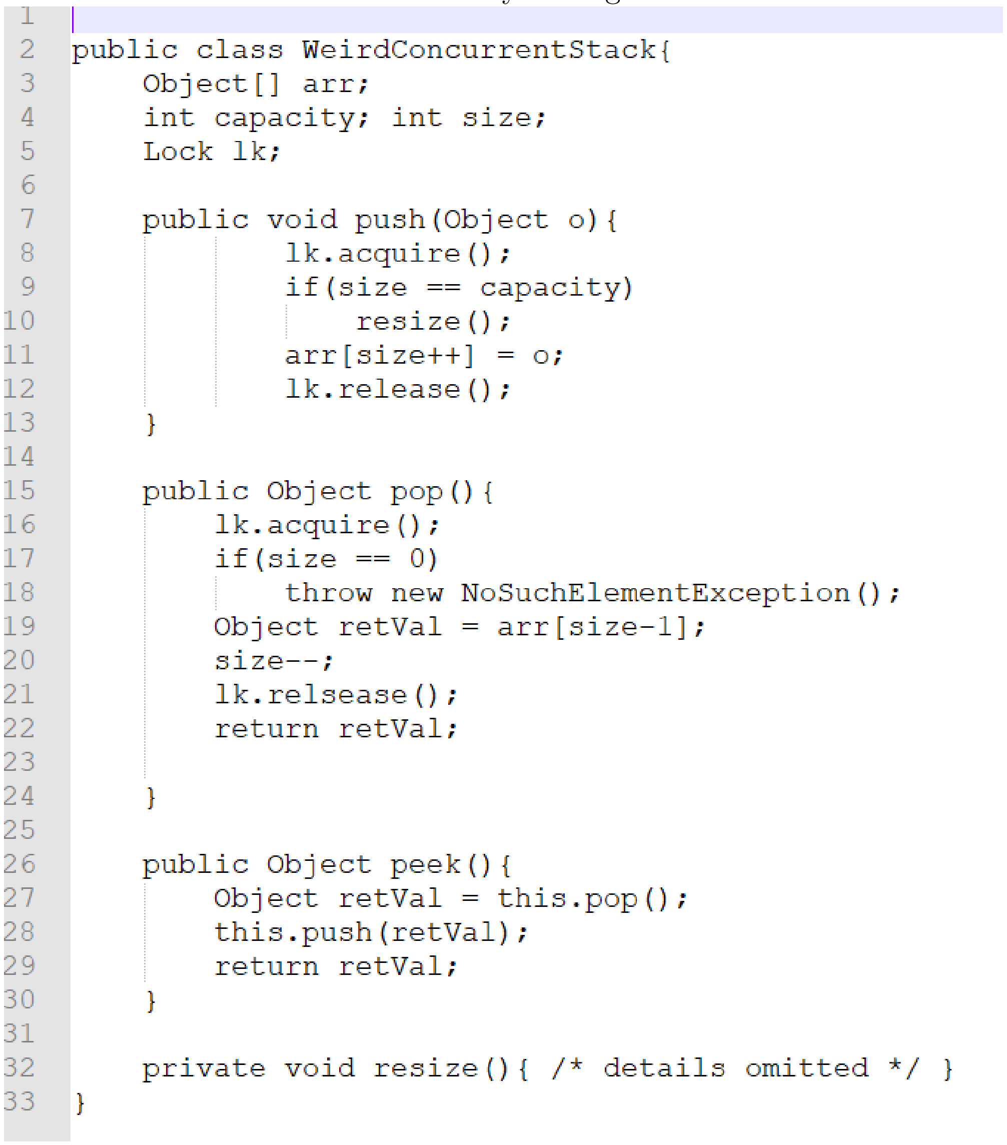
Your friend has written the following (somewhat strange) code for a stack.



```java

public class WeirdConcurrentStack{
    Object[] arr;
    int capacity; int size;

    public void push(Object o){
        if(size == capacity)
            resize();
        arr[size++] = o;
    }

    public Object pop(){
        if(size == 0)
            throw new NoSuchElementException();
        Object retVal = arr[size-1];
        size--;

    }

    public Object peek(){
        Object retVal = this.pop();
        this.push(retVal);
        return retVal;
    }

    private void resize(){ /* details omitted */ }
}
```

Give a bad interleaving of this code.

Your friend decides to fix the code by adding some re-entrant locks.



```java

public class WeirdConcurrentStack{
    Object[] arr;
    int capacity; int size;
    Lock lk;

    public void push(Object o){
            lk.acquire();
            if(size == capacity)
                resize();
            arr[size++] = o;
            lk.release();
    }

    public Object pop(){
        lk.acquire();
        if(size == 0)
            throw new NoSuchElementException();
        Object retVal = arr[size-1];
        size--;
        lk.relsease();
        return retVal;

    }

    public Object peek(){
        Object retVal = this.pop();
        this.push(retVal);
        return retVal;
    }

    private void resize(){ /* details omitted */ }
}
```

Does the code above have a bad interleaving? If so give a bad interleaving. If not informally justify why there won't be any.

Does the code above have potential for deadlock? If so describe an interleaving to cause deadlock. If not informally justify why deadlock will not occur.

Tell your friend a way to improve their code. If you found errors in the previous parts, your alterations should fix them. If you did not find errors, you should still find a way to improve the use of synchronization.

7 Sort

1. Robbie's first phone from middle school didn't have a lot of memory on it. How should his phone contacts be sorted to put them in last-name, first-name order?

2. MatLab (a mathematical programming language) implements some of its functions very quickly, on the assumption that the data is mostly sorted already. What kind of sort are they probably using?

3. Caitlin wants to run Radix sort on her extensive collection of books, sorting them alphabetically with their whole book contents as the input She wants to know not only if she has multiple copies of something, but also to see if differing versions have different cotents. Is this a good idea? Explain why either way. If it's not a good idea, recommend an input or sort that might work better.

8 Amdahl

1. Your company has a program which is 1/8 sequential and 7/8 parallelized. At a minimum, how many processors do you need for a 4x speedup? For full credit, your answer must be a simplified fraction or an integer.

2. How much of the program would have to be parallelized if you only have four processors?

9 P vs. NP

1. For each of the following problems, circle all classes which the problem is **known** to belong.

Determine if there is any pair of vertices in a graph that are at distance at least k from each other.	P	NP	NP-complete	NP-hard
Find the shortest tour of a weighted graph	P	NP	NP-complete	NP-hard
Determine if a graph has a topological sort	P	NP	NP-complete	NP-hard

2. For each of the following statements, say whether it is true or false, and justify your answer in 1-2 sentences.

   (a) If we found an efficient algorithm for 2-SAT, we would have a polynomial time algorithm for every problem in NP.

   (b) If we find an efficient algorithm for TSP, we would have a polynomial time algorithm for every NP-hard problem.