Final Exam

Autumn 2023

Name Answer Key

Net ID _____

Academic Integrity: You may not use any resources on this exam except for writing instruments, your own brain, and the exam packet itself. This exam is closed notes, closed neighbor, closed electronic devices, etc.. The last two pages of this exam provide a list of potentially helpful identities as well as room for scratch work (respectively). No markings on these last two pages will be graded. Your answer for each question must fit in the answer box provided.

Instructions: Before you begin, **Put your name and UW Net ID at the top of this page.** Make sure that your name and ID are LEGIBLE. Please ensure that all of your answers appear within the boxed area provided.

Section	Max Points
Asymptotic Analysis	12
Pre-Midterm Data Structures	12
Hash Tables	12
Sorting	10
Graphs	20
Parallelism	12
Concurrency	12
P vs NP	5
Extra Credit	(+2)
Total	95

Section 1: Asymptotic Analysis

(4 pts)**Question 1: Asymptotic Analysis of Code**

Give a simplified Θ bound on the best and worst case running times for the given code. (By simplified we mean it should contain no constant coefficients or non-dominant terms.) You should assume that no strings in the input list are null.

```
int doStuff(List<Integer> numbers){
    int n = numbers.size();
    int[] myList = {1,2,3,4};
    int i = 0;
    while(i < n){
        for (int j = 0; j < myList.length; j++){
            if(numbers[i] == myList[j]){
               i += n/2;
            }
        }
        i++;
    }
    return i;
}
```

(a) Best Case: $\Theta($ 1 $)$

(b) Worst Case: $\Theta($ n $)$

(8 pts)**Question 2: Which is larger?**

For each pair of functions $f(n)$ and $g(n)$ below, select the choice which best characterizes the asymptotic relationship between f and g. Write the letter corresponding with your answer in the box provided.

1. $f(n) = n$, $g(n) = \log_2(n)$

   A. $f(n) \in \Theta(g(n))$
   B. $f(n) \in O(g(n))$ and $f(n) \notin \Theta(g(n))$
   C. $f(n) \in \Omega(g(n))$ and $f(n) \notin \Theta(g(n))$

   C

2. $f(n) = \log_2(n^3)$, $g(n) = \log_5(n^5)$

   A. $f(n) \in \Theta(g(n))$
   B. $f(n) \in O(g(n))$ and $f(n) \notin \Theta(g(n))$
   C. $f(n) \in \Omega(g(n))$ and $f(n) \notin \Theta(g(n))$

   A

3. $f(n) = 2f(n/2) + n$, $g(n) = n^2$

   A. $f(n) \in \Theta(g(n))$
   B. $f(n) \in O(g(n))$ and $f(n) \notin \Theta(g(n))$
   C. $f(n) \in \Omega(g(n))$ and $f(n) \notin \Theta(g(n))$

   B

4. $f(n) = f(n-1) + 1$, $g(n) = g(n/2) + n$

   A. $f(n) \in \Theta(g(n))$
   B. $f(n) \in O(g(n))$ and $f(n) \notin \Theta(g(n))$
   C. $f(n) \in \Omega(g(n))$ and $f(n) \notin \Theta(g(n))$

   A

Creative Commons BY-NC 4.0

Section 2: Pre-Midterm Data Structures

(4 pts)**Question 3: Heap**

The array given below does not currently satisfy the heap property for a binary minheap, but calling percolate up/down on one item would result in a valid binary minheap.

The empty space below is available if you would like to draw the heap as a binary tree. This drawing is optional and it will not be graded.

7	12	9	15	11	11	22	16	18	35	14	31				
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

1. Give the index such that, if we called percolate **up** on that index, the result would be a valid heap.

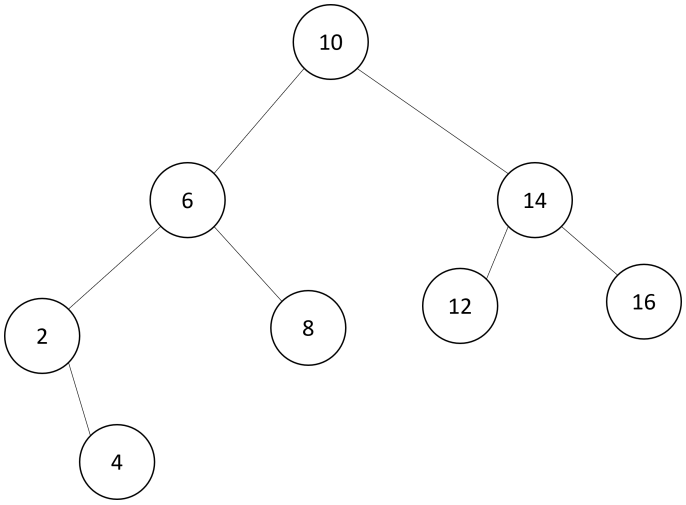
   4

2. Give the index such that, if we called percolate **down** on that index, the result would be a valid heap.

   1

Creative Commons BY-NC 4.0

(4 pts)**Question 4: AVL Tree**
Answer the following questions about the AVL Tree below. Each question should be considered completely independently (i.e. "reset" to the image between questions).



1. Give an integer key which, when inserted into the given AVL tree, would cause a double rotation.
   3
2. Give an integer key which, when inserted into the original AVL tree given above, would cause a single rotation. 5

(4 pts)**Question 5: B Tree**
Suppose we had a B Tree with $L = 20$ and $M = 10$ that has height 2. (Recall that we define height to be the maximum number of edges between the root and a leaf)

1. What is the minimum number of items in the data structure?
   100
2. What is the maximum number of items in the data structure?
   2000

Creative Commons BY-NC 4.0

Section 3: Hash Tables

(4 pts)**Question 6: Load Factor**

In 1-2 sentences, explain why, regardless of probing strategy, an open addressing hash table *must* be rehashed before its load factor exceeds 1.

A fully associative hash table cannot have a load factor greater than 1 because there cannot be more than 1 item stored at each index.

(4 pts)**Question 7: Quadratic Probing**

Insert 7, 66, 82, 67, 39, 10, 28 (in that order) into the open addressing hash table below. You should use the primary hash function $h(k) = k\%10$. In the case of collisions, use quadratic probing for collision resolution. If an item cannot be inserted into the table, please indicate this and continue inserting the remaining values.

Items that could not be inserted:

0	10
1	
2	82
3	
4	28
5	
6	66
7	7
8	67
9	39

Creative Commons BY-NC 4.0

(4 pts)**Question 8: Double Hashing**

Insert 7, 66, 82, 67, 39, 10, 28 (in that order) into the open addressing hash table below. You should use the primary hash function $h(k) = k\%10$. In the case of collisions, use double hashing for collision resolution where the secondary hash function is $g(k) = 1 + (k\%3)$. If an item cannot be inserted into the table, please indicate this and continue inserting the remaining values.

Items that could not be inserted:

0	39
1	
2	82
3	
4	10
5	
6	66
7	7
8	28
9	67

Creative Commons BY-NC 4.0

Section 4: Sorting

(4 pts)**Question 9: Sort Suggestions Hotline**

Suppose you are responsible for a phone-in service where people call in, describe the items they need to sort, and you recommend an algorithm for them to use. For each scenario described below, indicate which sorting algorithm property the caller would most need, then identify the *fastest* sorting algorithm which possesses that property from the options provided.

1. I am in charge of selling the new Taylor Swift Album. Instead of setting a price up front, I set up a web page to collect bids for a copy of the album, and in all I got too many bids to ship at once! I've decided to ship the copies in order of highest-bid-first. In the case that two people bid the same amount, I'm going to first ship to whomever entered their bid first. Currently the bids are listed in the order that I received them, so I need to figure out how to re-order them all into the proper shipping order. Should I use *Heap Sort*, *Merge Sort*, *Quick Sort*, or *Insertion Sort*?

   Property: Stable Sort

   Algorithm: Merge Sort

2. I'm working on a database that will maintain a constantly sorted list of objects. This list will grow as we receive more items from users. It is critical that the list is quickly updated and is always in sorted order. If we get a batch of new objects in quick succession we *must* start sorting as soon as we get the first object, we can't wait for all the objects in the batch arrive. I was thinking I would use *Merge Sort* so I could "merge" new elements in, but my friend thought I should "insert" them in instead which would suggest *Insertion Sort*.

   Property: Online (or Adaptive)

   Algorithm: Insertion Sort

(2 pts)**Question 10: Adaptive Sort**

What does it mean for a sorting algorithm to be adaptive? Write the letter of your choice in the box provided.

A. It does not use any auxiliary data structures.

B. It rearranges elements randomly to achieve better performance.

C. It runs faster if the list is close to sorted.

D. It can adapt to any data types in the input.

E. The relative order of "tied" elements is maintained.

C

(4 pts)**Question 11: Quick Sort Runtimes**
Answer each question in the box provided.

1. What is the best case runtime of quicksort? $\Theta(n \log n)$

2. Explain the context in which this best case runtime would be achieved (1-2 sentences).

   The pivot is consistently approximately the largest/smallest item in the list. I.e. the partition is consistently very unbalanced.

3. What is the worst case runtime of quicksort? $\Theta(n^2)$

4. Explain the context in which this runtime would be achieved (1-2 sentences).

   The pivot is near the middle of the list. I.e. the partition is balanced.

Creative Commons BY-NC 4.0

Section 5: Graphs

(8 pts)**Question 12: Maximum number of edges**
Below we will give several descriptions of a graph. In the box provided, indicate the maximum number of edges that graph could have.

1. A simple, directed graph with 10 nodes

   90

2. A simple, undirected graph with 10 nodes

   45

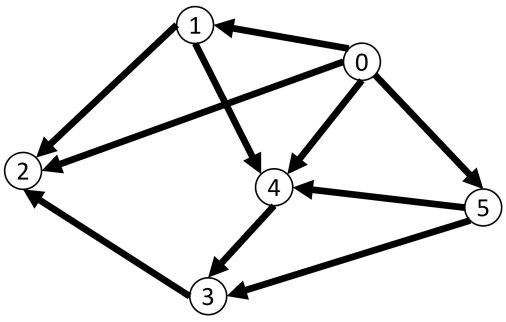
3. A simple, undirected graph with 10 nodes where each node's degree is not more than 1

   5

4. A Tree with 10 nodes

   9

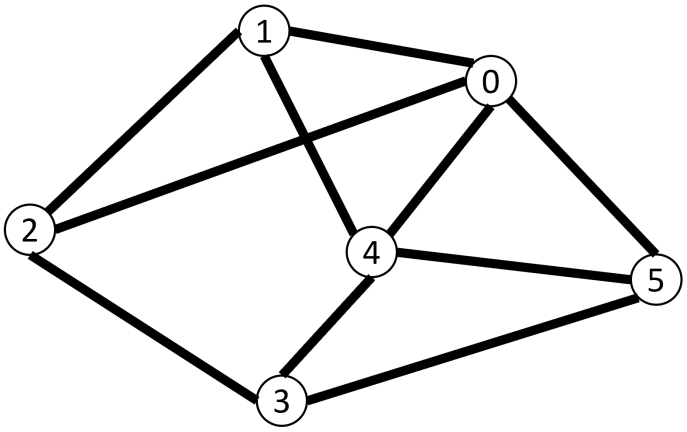
(4 pts)**Question 13: Topological Sort**
List the nodes in the graph below such that they are in two different topologically sorted orderings.



Topological Order 1: 0,5,1,4,3,2

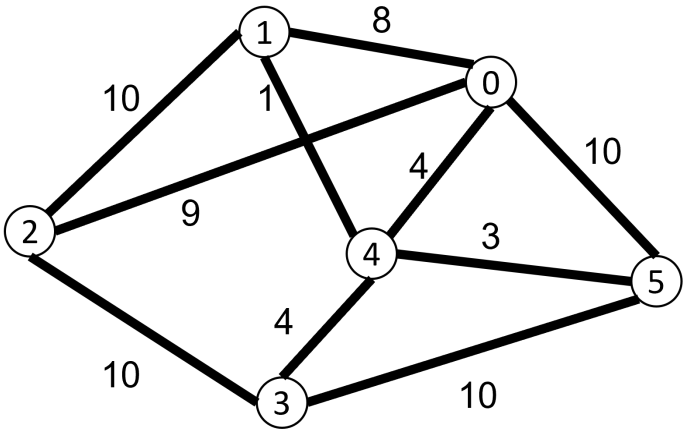
Topological Order 2: 0,1,5,4,3,2

(2 pts)**Question 14: BFS**
For the graph below, list the nodes an order that they might be removed from the queue in a BFS starting from node 0 (note that this is the same as the graph above, but now undirected).



BFS Order: 0,1,2,4,5,3

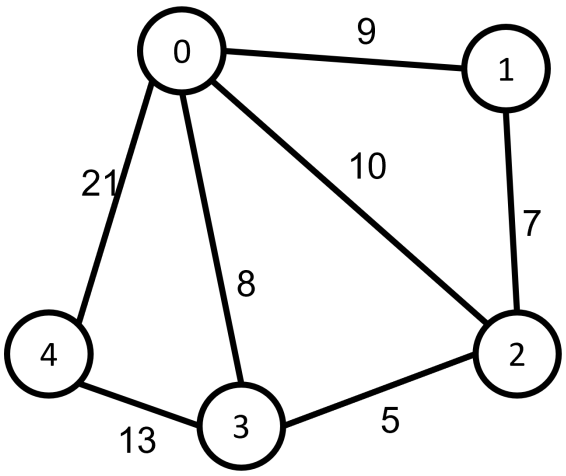
(2 pts)**Question 15: Dijkstras**
For the graph below, list the nodes an order that they might be removed from the priority queue when running Dijkstra's algorithm starting from node 0 (note that his is the same as the previous graph, but now with weights).



Dijkstra's Order: 0,4,1,5,3,2

Creative Commons BY-NC 4.0

(6 pts)**Question 16: MSTs**
The next 2 questions will relate to running minimum spanning tree algorithms (Kruskal's and Prim's) on the graph below:



1. What are the weights of the first three edges added to the minimum spanning tree when running Kruskal's algorithm?

   First edge's weight: 5

   Second edge's weight: 7

   Third edge's weight: 8

2. What are the first three edges added to the minimum spanning tree when running Prims's algorithm starting with node 0?

   First edge's weight: 8

   Second edge's weight: 5

   Third edge's weight: 7

Section 6: Parallelism

(8 pts)**Question 17: ForkJoin**

For the next series of questions you will write code to implement the following sequential algorithm as a parallel algorithm using the Java ForkJoin Framework.

The following sequential method finds the sum of all odd values in the given array.

```
int sumOdd(in[] arr){
    int sum = 0;
    for(int i = 0; i < arr.length; i++{
        if (arr[i]%2==1)
            sum += arr[i];
    }
    return sum;
}
```

On the next page we have provided the majority of the code to implement `sumOdd` in parallel using ForkJoin and RecursiveTask. In particular, we have provided a main class which invokes the ForkJoin Pool, hve written the constructor for the RecursiveTask class, and have provided the code that will run when the array length is below the sequential cutoff (we chose 100 as the cutoff) insdie the compute method. Complete our implementation by finishing the compute method.

Creative Commons BY-NC 4.0

```
import java.util.concurrent.ForkJoinPool;
import java.util.concurrent.RecursiveTask;

public class Main {
    public static final ForkJoinPool fjPool = new ForkJoinPool();
    public static Integer SumOdd (int[] input) {
        return fjPool.invoke(new SumOddTask(input, 0, input.length));
    }
}

public class SumOddTask extends RecursiveTask<Integer> {
    int[] arr;
    int hi;
    int lo;

    public SumOddTask(int[] arr, int lo, int hi){
        this.arr = arr;
        this.hi = hi;
        this.lo = lo;
        this.sum = 0;
    }

    public int compute(){
        if (hi-lo < 100){
            for(int i = lo; i < hi; i++){
                if (arr[i]%2==1)
                    this.sum += arr[i];
            }
            return this.sum;
        }
        \\ finish the compute method here
```

SumOddTask left = new SumOddTask(arr,lo,(hi+lo)/2);
SumOddTask right= new SumOddTask(arr,(hi+lo)/2,hi);
left.fork();
int rightAns = right.compute();
int leftAns = left.join();
return leftAns + rightAns;

```
    }
```

(2 pts)**Question 18: Map,Reduction**
Express the task from the previous question(that of finding the sum of all odd values in an array) as a map and a reduction (use psuedocode or 1-2 sentences of English).

Map all values in the array to 0 if even (and themselves if odd), then use a reduction to find the sum of the array.

(2 pts)**Question 19: Parallel Pack**

Suppose after the first stage of a parallel pack (i.e. filter) operation (that is, the map applying a boolean function to each index of the input array), you find that the value of index i is 0, index $i+1$ is 1, and index $i+2$ is 1.

Next, suppose that after the second stage of the same parallel pack operation (that is, the parallel prefix sum), you find that the value of index i is 18.

1. At what index of the final output array will you find the value that was at index $i+1$ of the input array?

   18

2. In what index of the output array will you find the value that was at index $i+2$ of the input array?

   19

Creative Commons BY-NC 4.0

Section 7: Concurrency

Suppose two people are eating a meal, but they must share one set of cutlery (fork and knife). A meal consists of a salad and an entree. To eat some salad requires just the fork, to eat some entree requires both the fork and the knife. Washing the cutlery requires both the fork and the knife. Below we have java classes that manage the sharing. Each question in this section will relate to these java classes provided. In each case every thread will have its own local Meal object, but there will only be one instance of the Cutlery class shared among all threads. Line numbers have been provided for reference in the upcoming questions.

```
1       public class Meal{
2           public int saladLeft = 10;
3           public int entreeLeft = 10;
4       }
5
6       public class Cutlery{
7           Object fork = new Object();
8           Object knife = new Object();
9           boolean dirty = false;
10
11          public void eatSalad(Meal m){
12              synchronized(fork){
13                  m.saladLeft--;
14                  this.dirty = true;
15              }
16          }
17
18          public void eatEntree(Meal m){
19              synchronized(fork){
20                  synchronized(knife){
21                      m.entreeLeft--;
22                      this.dirty = true;
23                  }
24              }
25          }
26
27          public void wash(){
28              if(this.dirty){
29                  synchronized(knife){
30                      synchronized(fork){
31                          System.out.println("Washing!");
32                          this.dirty = false;
33                      }
34                  }
35              }
36          }
37      }
```

(2 pts)**Question 20: Data Race**
The above code contains a data race. Describe the data race using 1-2 sentences.

If two threads both call wash then there can be a simultaneous read in the if statement check and write in the body.

(2 pts)**Question 21: Deadlock**
The above code also contains a potential for deadlock. Describe a situation in which deadlock would occur (1-2 sentences).

If T1 calls eatEntree while T2 calls wash then T2 could lock the knife, then T1 could lock the fork.

Creative Commons BY-NC 4.0

(8 pts)**Question 22: Deadlock and/or Race Condition**

Below we provide alternative implementations of the wash method. For each, indicate whether that version of wash (along with the rest of the code above) has a potential for deadlock and whether it contains a race condition by writing "yes" or "no" in the corresponding box.

1.
```
synchronized public void wash(){
    if (this.dirty){
        System.out.println("Washing!");
        this.dirty = false;
    }
}
```

Deadlock? No

Race Condition? Yes

2.
```
public void wash(){
    synchronized(knife){
        synchronized(fork){
            if (this.dirty){
                System.out.println("Washing!");
                this.dirty = false;
            }
        }
    }
}
```

Deadlock? Yes

Race Condition? No

3.
```
public void wash(){
    synchronized(knife){
        Meal m = new Meal();
        this.eatSalad(m);
        if(this.dirty){
            System.out.println("Washing!");
            this.dirty = false;
        }
    }
}
```

Deadlock? Yes

Race Condition? Yes

4.
```
synchronized void wash(){
    synchronized(knife){
        synchronized(fork){
            if (this.dirty){
                System.out.println("Washing!");
                this.dirty = false;
            }
        }
    }
}
```

Deadlock? Yes

Race Condition? No

Section 8: P, NP, NP-Hard, NP-Complete

(5 pts)**Question 23: Am I a millionaire?**

For each scenario below we would be able to conclude one of:

A. $\mathsf{P} = \mathsf{NP}$,

B. $\mathsf{P} \subset \mathsf{NP}$,

C. *Inconclusive* (meaning that scenario alone does not resolve the question).

Correctly identify which conclusion we can draw from each scenario by writing the corresponding letter in the box provided.

1. A problem from EXP is found to belong to $\mathsf{NP-Complete}$.

   C

2. The Independent Set Problem is found to be outside of P.

   B

3. The Minimum Spanning Tree problem is found to be in class NP.

   C

4. The problem of finding the shortest path in a graph with negative-weight edges is found to be in class $\mathsf{NP-Complete}$.

   A

5. The problem of determining whether a given graph has a Hamiltonian Path is found to belong to class P.

   A

Creative Commons BY-NC 4.0

Extra Credit

(2 pts)**Question Extra Credit: What did you learn this quarter?**
Nathan's grandmother, Elouise, is 94 years old and has never used a computer before in her life. Summarize what you learned this quarter in a way that Elouise would appreciate.

text

Creative Commons BY-NC 4.0

Scratch Work

Nothing written on this page will be graded.

Creative Commons BY-NC 4.0

Identities

Nothing written on this page will be graded.

Summations

$$\sum_{i=0}^{\infty} x^i = \frac{1}{1-x} \text{ for } |x| < 1$$

$$\sum_{i=0}^{n-1} = \sum_{n}^{i=1} = n$$

$$\sum_{i=0}^{n} i = 0 + \sum_{n}^{i=1} i = \frac{n(n+1)}{2}$$

$$\sum_{i=1}^{n} i^2 = \frac{n(n+1)(2n+1)}{6} = \frac{n^3}{3} + \frac{n^2}{2} + \frac{n}{6}$$

$$\sum_{i=0}^{n} i^3 = \left(\frac{n(n+1)}{2}\right)^2 = \frac{n^4}{4} + \frac{n^3}{2} + \frac{n^2}{4}$$

$$\sum_{i=0}^{n-1} x^i = \frac{1-x^n}{1-x}$$

$$\sum_{i=0}^{n-1} \frac{1}{2^i} = 2 - \frac{1}{2^{n-1}}$$

Logs

$$x^{\log_x(n)} = n$$

$$\log_a(b^c) = c \log_a(b)$$

$$a^{\log_b(c)} = c^{log_b(a)}$$

$$\log_b(a) = \frac{\log_d(a)}{\log_d(b)}$$

Creative Commons BY-NC 4.0