HW1 Distribution
HW2 Distribution
HW3 Distribution

![Histogram of HW3 Distribution]
P1 Distribution

![Graph showing P1 Distribution](image-url)
P2A Distribution
Midterm Distribution
Crudely Estimated Grades
(out of 39 points so far)
1) 10 Points

**List ADT**
Name two potential implementations of the List ADT that were discussed in class, and provide $O(\log n)$ bounds on the running time of each operation.

<table>
<thead>
<tr>
<th>Name</th>
<th>Insert Average</th>
<th>Find Average</th>
<th>Delete Average</th>
</tr>
</thead>
</table>

**Priority Queue ADT**
Name an implementation of the Priority Queue ADT that was discussed in class and has the desired property. Provide $O(\log n)$ bounds on the running time of each operation.

<table>
<thead>
<tr>
<th>Name</th>
<th>Insert Average</th>
<th>Delete Min Average</th>
<th>Merge Average</th>
</tr>
</thead>
</table>

**Dictionary ADT**
Name an implementation of the Dictionary ADT that was discussed in class and has the desired property. Provide $O(\log n)$ bounds on the running time of each operation.

<table>
<thead>
<tr>
<th>Name</th>
<th>Average Insert</th>
<th>Worst Case Insert</th>
<th>Worst-Case Name of k Inserts</th>
</tr>
</thead>
</table>

Conceptually Simple
Efficient
Worst-Case Guarantee
Efficient Amortized Costs
Better Addressed Costs of Cache and Disk Access
Q1 Distribution
2) 10 Points

Compute an approximate upper bound $O(2^n n^2)$ based on the recurrence time of each code fragment. Circle your answer for each code fragment.

a) 
```c
for (i = 0; i < n; i++) {
    \text{compute} \ldots
```

b) 
```c
for (i = 0; i < n; i++) {
    \text{for (j = 0; j < i; j++)} {
        \text{compute} \ldots
    }
    \text{compute} \ldots
```

c) 
```c
for (i = 0; i < n; i++) {
    \text{for (j = 0; j < i; j++)} {
        \text{compute} \ldots
    }
    \text{count} \ldots
    \text{for (k = 0; k < i; k++)} {
        \text{compute} \ldots
    }
}
```

d) 
```c
for (i = 0; i < n; i++) {
    \text{for (j = 0; j < i; j++)} {
        \text{compute} \ldots
    }
    \text{count} \ldots
    \text{for (k = 0; k < i; k++)} {
        \text{compute} \ldots
    }
}
```
Q2 Distribution
Problem 3

3. 10 Points

Prove by induction that

\[ 1 + 4 + 9 + \cdots + n^2 = \frac{n(n+1)(2n+1)}{6} \]

for every positive integer \( n \).

If you find yourself unable to factor a polynomial, recall that you know what the polynomial should factor to. Determining what the factors should be and showing that the result is equal to your polynomial would be appropriate.
Q3 Distribution
4) 10 Points

A level-order traversal visits each node in a tree according to the depth of that node. For example, the nodes in this binary tree:

```
    6
   / \
  7   5
 / \ / \ (depth 3)
2  4 9
```

would be visited in the order $6, 5, 7, 2, 4, 9$. The in-order, pre-order, and post-order traversals described in class are based on an implicit stack, created by the recursion. This task is intended to illustrate that you cannot perform a level-order traversal using recursion.

You will write pseudocode that uses a queue to print nodes in a binary tree according to a level-order traversal.

a) Define an appropriate node type (storing an int as the data element in each node):

```java
class Node {
  int data;
  Node left, right;
}
```

b) Define an appropriate interface for a node queue (no need to worry about elements):

```java
interface NodeQueue {
  void add(Node node);
  Node poll();
}
```
Q4 Distribution
Consider the following binary heap:

```
     3
    / \
   5   4
  / \ / \
3 7 1 5 6
```

Perform the following operations in order, drawing the resulting heap after each operation and noting it as the starting point for the next operation. You need only show the result of each operation for full credit, but we will only be able to award partial credit if you show your work. If the space available here is insufficient, write the back of this sheet (clearly labeling your work). Circle the state of the heap after each operation so that we can distinguish it from your intermediate work.

a) Insert 25

b) Insert 2

c) Delete Min

d) Delete Min

e) Delete Min
Q5 Distribution
6) 5 Points

Consider the following binomial queue, currently containing 8 trees:

Perform a deleteMin operation on this binomial queue. You need only show the result of the operation for full credit, but we will only be able to award partial credit if you show your work. If the space available here is insufficient, see the back of this sheet (clearly labeling your work). Circle the final state of the queue so that we can distinguish it from your intermediate work.
Q6 Distribution
Consider the following AVL tree:

```
    4
   / \
  1   5
 / \ / \
0  2 3  6
```

Perform the following operations in order, drawing the resulting tree after each operation and using it as the starting point for the next operation. You need only show the result of each operation for full credit, but we will only be able to award partial credit if you show your work. If the space available here is insufficient, use the back of this sheet (clearly label your work). Circle the state of the tree after each operation so that we can distinguish it from your intermediate work.

a) Insert 8

b) Insert 2

c) Insert 6

d) Delete 1 (We did not cover general AVL deletion, but you should know how to perform this delete)
Q7 Distribution
b) 10 Points

Consider the development of a B-tree (specifically, the variant of B-tree discussed in class and in Chapter 4). Assume that the machine your B-tree will run on stores and retrieves data in blocks of 1024 bytes and that storing a reference requires 4 bytes. Assume that you are not storing null values.

You will be storing personal information indexed by a person’s Social Security Number (SSN). An SSN is stored in 9 bytes. The exact information being stored about each person is unknown but requires a total of 125 bytes (including space for the SSN identifying the record).

You should provide an accurate answer for questions a, b, and c. Just providing the formula you used to obtain your answer will allow us to give full or partial credit if your answer is incorrect due to a simple arithmetic error.

a) Choose an appropriate value for M, the branching factor of internal nodes.

b) Choose an appropriate value for L, the number of elements per leaf.

c) What is the maximum number of records your B-tree could contain if its height is 2?

d) In terms of M, L, and h, what is the maximum number of records a B-tree of height h might contain?

Extra Credit: In terms of M, L, and h, what is the minimum number of records a B-tree of height h must contain?
Q8 Distribution