This test has 8 questions worth a total of 60 points. You have 80 minutes. The exam is closed book, except that you are allowed to use a one page cheatsheet. No calculators or other electronic devices are permitted. Give your answers and show your work in the space provided. Write out and sign the Honor Code pledge before turning in the test.

“I pledge my honor that I have not violated the Honor Code during this examination.”
0. Miscellaneous. (2 point)

In the space provided on the front of the exam, write your name and Princeton NetID; circle your precept number; and sign the honor code.

1. Analysis of algorithms. (14 points)

(a) For each expression in the left column, give the best matching description from the right column.

\[ 1 + 2 + 4 + 8 + 16 + \ldots + N \quad \text{A. } \sim \frac{1}{2}N^2. \]
\[ 1 + 2 + 3 + 4 + 5 + \ldots + N \quad \text{B. } O(N^2). \]
\[ 1 + 3 + 5 + 7 + 9 + \ldots + N \quad \text{C. Both A and B.} \]
\[ \frac{1}{2}N^2 \quad \text{D. Neither A nor B.} \]
\[ \frac{1}{2}N^2 + 100N \lg N \]
\[ N^2 \]
\[ N^3 \]

(b) For each quantity in the left column, give the best matching description from the right column.

\[ \text{Height of a weighted quick union data structure with } N \text{ items.} \quad \text{A. } \sim \lg N \text{ in the best case.} \]
\[ \text{Height of a binary heap with } N \text{ keys.} \quad \text{B. } \sim \lg N \text{ in the worst case.} \]
\[ \text{Height of a left-leaning red-black BST with } N \text{ keys.} \quad \text{C. Both A and B.} \]
\[ \text{Maximum function-call stack size when (top-down) mergesorting } N \text{ keys.} \quad \text{D. Neither A nor B.} \]
\[ \text{Maximum function-call stack size when quicksorting } N \text{ keys.} \]
\[ \text{Number of compares to binary search in a sorted array of size } N. \]
(c) Consider the following code fragment.

```java
int count = 0;
int N = a.length;
Arrays.sort(a);
for (int i = 0; i < N; i++) {
    for (int j = i+1; j < N; j++) {
        if (Arrays.binarySearch(a, a[i] + a[j]) count++;
    }
}
```

Suppose that it takes 1 second when $N = 3,500$. Approximately how long will it take when $N = 35,000$? Circle the best answer.

10 seconds  20 seconds  1 minute  2 minutes  1 hour  2 hours

(d) Consider the following Java data type definition for a 2-3 tree, where the nested class `Node` represents either a 2-node or a 3-node.

```java
public class TwoThreeTree<Key extends Comparable<Key>, Value> {
    private Node root;

    private class Node {
        private int count; // subtree count
        private Key key1, key2; // the one or two keys
        private Value value1, value2; // the one or two values
        private Node left, middle, right; // the two or three subtrees
    }

    ...
}
```

How much memory (in bytes) does each `Node` object consume? Circle your answer.
2. 8 sorting algorithms. (8 points)

The column on the left is the original input of strings to be sorted; the column on the right are the string in sorted order; the other columns are the contents at some intermediate step during one of the 8 sorting algorithms listed below. Match up each algorithm by writing its number under the corresponding column. Use each number exactly once.

<table>
<thead>
<tr>
<th>john</th>
<th>aviv</th>
<th>alan</th>
<th>ctaï</th>
<th>andy</th>
<th>aviv</th>
<th>yort</th>
<th>alan</th>
<th>anna</th>
<th>alan</th>
</tr>
</thead>
<tbody>
<tr>
<td>tžha</td>
<td>azhu</td>
<td>anna</td>
<td>fyau</td>
<td>alan</td>
<td>azhu</td>
<td>tžha</td>
<td>andy</td>
<td>fyau</td>
<td>andy</td>
</tr>
<tr>
<td>fyau</td>
<td>ctaï</td>
<td>fyau</td>
<td>john</td>
<td>ddix</td>
<td>ctaï</td>
<td>vyas</td>
<td>anna</td>
<td>ctaï</td>
<td>anna</td>
</tr>
<tr>
<td>ctaï</td>
<td>ddix</td>
<td>ctaï</td>
<td>tžha</td>
<td>azhu</td>
<td>ddix</td>
<td>ravi</td>
<td>aviv</td>
<td>andy</td>
<td>aviv</td>
</tr>
<tr>
<td>nbal</td>
<td>fyau</td>
<td>andy</td>
<td>azhu</td>
<td>azhu</td>
<td>fyau</td>
<td>sida</td>
<td>azhu</td>
<td>ddix</td>
<td>azhu</td>
</tr>
<tr>
<td>ddix</td>
<td>john</td>
<td>ddix</td>
<td>ddix</td>
<td>anna</td>
<td>john</td>
<td>oleg</td>
<td>azhu</td>
<td>azhu</td>
<td>azhu</td>
</tr>
<tr>
<td>sguo</td>
<td>kuan</td>
<td>azhu</td>
<td>nbal</td>
<td>fyau</td>
<td>kuan</td>
<td>sguo</td>
<td>ctaï</td>
<td>azhu</td>
<td>ctaï</td>
</tr>
<tr>
<td>azhu</td>
<td>nbal</td>
<td>azhu</td>
<td>sguo</td>
<td>ctaï</td>
<td>nbal</td>
<td>peck</td>
<td>ddix</td>
<td>aviv</td>
<td>ddix</td>
</tr>
<tr>
<td>aviv</td>
<td>sguo</td>
<td>aviv</td>
<td>aviv</td>
<td>john</td>
<td>oleg</td>
<td>ctaï</td>
<td>fyau</td>
<td>alan</td>
<td>fyau</td>
</tr>
<tr>
<td>sida</td>
<td>sida</td>
<td>john</td>
<td>kuan</td>
<td>aviv</td>
<td>sguo</td>
<td>nbal</td>
<td>john</td>
<td>john</td>
<td>john</td>
</tr>
<tr>
<td>kuan</td>
<td>tžha</td>
<td>kuan</td>
<td>sida</td>
<td>kuan</td>
<td>sida</td>
<td>kuan</td>
<td>kuan</td>
<td>vyas</td>
<td>kuan</td>
</tr>
<tr>
<td>vyas</td>
<td>vyas</td>
<td>vyas</td>
<td>vyas</td>
<td>lily</td>
<td>tžha</td>
<td>lily</td>
<td>kwak</td>
<td>oleg</td>
<td>kwak</td>
</tr>
<tr>
<td>oleg</td>
<td>kwak</td>
<td>oleg</td>
<td>kwak</td>
<td>nbal</td>
<td>vyas</td>
<td>fyau</td>
<td>oleg</td>
<td>levý</td>
<td>levý</td>
</tr>
<tr>
<td>levý</td>
<td>levý</td>
<td>levý</td>
<td>levý</td>
<td>kwak</td>
<td>levý</td>
<td>levý</td>
<td>levý</td>
<td>kwak</td>
<td>lily</td>
</tr>
<tr>
<td>kwak</td>
<td>muir</td>
<td>kwak</td>
<td>muir</td>
<td>sguo</td>
<td>kwak</td>
<td>kwak</td>
<td>vyas</td>
<td>muir</td>
<td>muir</td>
</tr>
<tr>
<td>muir</td>
<td>oleg</td>
<td>muir</td>
<td>oleg</td>
<td>muir</td>
<td>muir</td>
<td>muir</td>
<td>muir</td>
<td>peck</td>
<td>nbal</td>
</tr>
<tr>
<td>peck</td>
<td>peck</td>
<td>peck</td>
<td>alan</td>
<td>oleg</td>
<td>peck</td>
<td>azhu</td>
<td>peck</td>
<td>ravi</td>
<td>oleg</td>
</tr>
<tr>
<td>ravi</td>
<td>ravi</td>
<td>ravi</td>
<td>peck</td>
<td>levý</td>
<td>ravi</td>
<td>aviv</td>
<td>ravi</td>
<td>kuan</td>
<td>peck</td>
</tr>
<tr>
<td>alan</td>
<td>alan</td>
<td>sida</td>
<td>ravi</td>
<td>sida</td>
<td>alan</td>
<td>alan</td>
<td>sida</td>
<td>yort</td>
<td>ravi</td>
</tr>
<tr>
<td>yort</td>
<td>andy</td>
<td>yort</td>
<td>yort</td>
<td>vyas</td>
<td>yort</td>
<td>john</td>
<td>yort</td>
<td>sida</td>
<td>sguo</td>
</tr>
<tr>
<td>azhu</td>
<td>anna</td>
<td>sguo</td>
<td>andy</td>
<td>peck</td>
<td>azhu</td>
<td>azhu</td>
<td>nbal</td>
<td>sguo</td>
<td>sida</td>
</tr>
<tr>
<td>andy</td>
<td>azhu</td>
<td>nbal</td>
<td>anna</td>
<td>ravi</td>
<td>andy</td>
<td>andy</td>
<td>tžha</td>
<td>nbal</td>
<td>tžha</td>
</tr>
<tr>
<td>anna</td>
<td>lily</td>
<td>tžha</td>
<td>azhu</td>
<td>tžha</td>
<td>anna</td>
<td>anna</td>
<td>sguo</td>
<td>lily</td>
<td>vyas</td>
</tr>
<tr>
<td>lily</td>
<td>yort</td>
<td>lily</td>
<td>lily</td>
<td>yort</td>
<td>lily</td>
<td>ddix</td>
<td>lily</td>
<td>tžha</td>
<td>yort</td>
</tr>
<tr>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
</tr>
<tr>
<td>0</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>1</td>
</tr>
</tbody>
</table>

(0) Original input  (1) Sorted  (2) Selection sort  (3) Insertion sort  (4) Shellsort  (5) Mergesort  (6) Mergesort  (7) Quicksort  (8) Quicksort  (9) Heapsort

Shellsort: (13-4-1 increments)
Mergesort: (top-down)
Mergesort: (bottom-up)
Quicksort: (standard, no shuffle)
Quicksort: (3-way, no shuffle)
Heapsort
3. Binary heaps. (6 points)

Consider the following binary tree representation of a max-heap.

(a) Give the array representation of the heap.

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(b) Delete the maximum key. Give the resulting heap, circling any entries that changed.

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(c) Insert the key Q into the original binary heap. circling any entries that changed.

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
4. **Red-black BSTs. (8 points)**

Consider the following left-leaning red-black BST.

(a) Which one or more of the keys below could be in the node labeled with a question mark?

A  B  C  D  E  F  G  H  I  J  K  L  M  N  O  P  Q  R  S  T  U  V  W  X  Y  Z

(b) What is the color of the link from the node labeled with a question mark to its parent?
Circle the best answer.

BLACK  RED  either RED or BLACK

(c) Add the key U and draw the resulting left-leaning red-black BST.
5. **Hashing. (6 points)**

Suppose that the following keys are inserted in the order

\[
A \ B \ C \ D \ E \ F \ G
\]

into an initially empty linear-probing hash table of size 7, using the following hash function:

<table>
<thead>
<tr>
<th>key</th>
<th>hash(key, 7)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>3</td>
</tr>
<tr>
<td>B</td>
<td>1</td>
</tr>
<tr>
<td>C</td>
<td>4</td>
</tr>
<tr>
<td>D</td>
<td>1</td>
</tr>
<tr>
<td>E</td>
<td>5</td>
</tr>
<tr>
<td>F</td>
<td>2</td>
</tr>
<tr>
<td>G</td>
<td>5</td>
</tr>
</tbody>
</table>

What is the result of the linear-probing array?

*Assume that the array size is fixed and does not double.*
6. Bitonic max. (8 points)

An array is *bitonic* if it consists of a strictly increasing sequence of keys immediately followed by a strictly decreasing sequence of keys. Design an algorithm that determines the maximum key in a bitonic array of size \( N \) in time proportional to \( \log N \).

(a) Give a crisp and concise English description of your algorithm—don’t write Java code. Your answer will be graded on correctness, efficiency, and clarity.

(b) To demonstrate your algorithm, list the first four compares that your algorithm performs to find the maximum key in the following bitonic array of 15 elements:

<table>
<thead>
<tr>
<th>i</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
</tr>
</thead>
<tbody>
<tr>
<td>a[i]</td>
<td>10</td>
<td>34</td>
<td>56</td>
<td>76</td>
<td>87</td>
<td>80</td>
<td>70</td>
<td>66</td>
<td>56</td>
<td>30</td>
<td>28</td>
<td>25</td>
<td>20</td>
<td>15</td>
<td>11</td>
</tr>
</tbody>
</table>
7. Stable priority queue. (8 points)

A min-based priority queue is stable if min() and deleteMin() return the minimum key that was least-recently inserted. Describe how to implement a StableMinPQ data type such that every operation takes at most (amortized) logarithmic time.

```
public class StableMinPQ<Key extends Comparable<Key>> {
    StableMinPQ() {
        create an empty priority queue
    }
    Key min() {
        return the minimum key
        (that was least-recently inserted)
    }
    Key delMin() {
        delete and return the minimum key
        (that was least-recently inserted)
    }
    void insert(Key key) {
        insert the key
    }
    boolean isEmpty() {
        is the priority queue empty?
    }
}
```

Give a crisp and concise English description of your algorithm—don’t write Java code. Your answer will be graded on correctness, efficiency, and clarity.