Data Structures
CSCI H343, Fall 2022 [A]

Midterm

Name: _____

This exam has 11 questions, for a total of 100 points.

1. [12 points] What is the output of running the `main` method of class `C`?

```
class Point {
    int x, y;
    Point(int x_, int y_) { x = x_; y = y_; }
}
class C {
    static int add(int x, int y) { x += y; return x; }
    static Point add(Point p, Point q) {
        p.x += q.x; p.y += q.y; return p;
    }
    static void println(int n) { System.out.println(n); }
    public static void main(String[] args) {
        int x = 3; int y = 4;
        int z = C.add(x,y);
        x = x + 1; y = y + 1;
        println(x); println(y); println(z);

        Point p = new Point(x,y); Point q = new Point(x,y);
        Point r = C.add(p, q);
        p.x = p.x + 1; q.x = q.x + 1;
        println(p.x); println(q.x); println(r.x);
    }
}
```

2. [10 points] Fill in the blanks to complete the following implementation of the `Sequence` and `Iter` interfaces using a Java array.

```
interface Sequence<T> {
    Iter<T> begin();
    Iter<T> end();
}
```

```
interface Iter<T> {
    T get();
    void advance();
    Iter<T> clone();
    boolean equals(Iter<T> other);
}
```

```
class Array<T> implements Sequence<T> {
    T[] data;
    public Array(T[] a) { ___(a)___ }
    public class ArrayIter implements ___(b)___ {
        int pos;
        ArrayIter(int p) { pos = p; }
        public T get() { return ___(c)___; }
        public void advance() { ++pos; }
        public Iter<T> clone() { return new ArrayIter(pos); }
        public boolean equals(Iter<T> other) {
            return ___(d)___;
        }
    }
    public Iter<T> begin() { return ___(e)___; }
    public Iter<T> end() { return new ArrayIter(data.length); }
}
```

3. [10 points] Implement the `find` method for the following `BinarySearchTree` class.

```
class BinarySearchTree<K> implements OrderedSet<K> {
    Node<K> root;
    int numNodes;
    BiPredicate<K, K> lessThan;

    /**
     * Finds the node with the given key in the subtree of node curr,
     * or if there is none, the parent of where such a node would be.
     * @param key
     * @param curr The current node.
     * @param parent The parent of the current node.
     */
    Node<K> find(K key, Node<K> curr, Node<K> parent) {
```

4. [8 points] Recall that $f \lesssim g$ if and only if $\exists kc.\forall n \geq k.f(n) \leq cg(n)$. Using this fact, prove that for any functions f and g, if $f \lesssim g$ then $f + g \lesssim g$.

5. 6 points Recall the definition of asympototically tight bound.

$$f \in \Theta(g) \text{ means } \exists k c_1 c_2, \forall n \geq k, 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n).$$

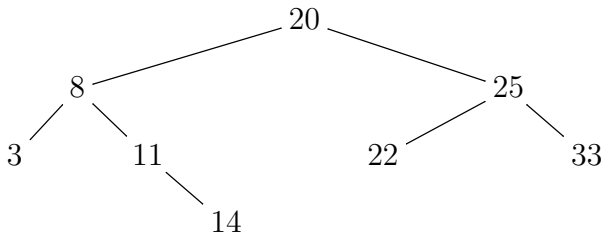
Give an asymptotically tight bound for the following code.

```
sum = 0;
for (int k = 1; k <= n; k = k * 2) {
  for (int j = 1; j <= n; ++j) {
    ++sum;
  }
}
```

6. 10 points The following `insert_sorted` method inserts the given `data` into a sorted linked list (sorted smallest-to-largest integer), producing a new sorted link list and leaving the original linked list unchanged. Fill in the blanks to complete `insert_sorted`.

```
class ListNode {
    int data;
    ListNode next;
    public ListNode(int val, ListNode next) {
        this.data = val; this.next = next;
    }
    public ListNode push(int val) {
      return new ListNode(val, this);
    }
    public ___(a)___ insert_sorted(int data) {
        if (data < this.data) {
            return this.push(___(b)___);
        } else if (this.next == null) {
            ListNode n = new ListNode(data, null);
            return n.push(___(c)___);
        } else {
            ListNode rest = ___(d)___;
            return rest.push(___(e)___);
        }
    }
}
```

7. [8 points] Given the following AVL binary search tree, insert key 12, maintaining the binary search tree and AVL properties by using left and/or right rotations. Explain which nodes violate the AVL property and explain each change that you make to the tree. Draw the tree after each change.



8. [8 points] Give the big-O time complexity for each of the following methods of the `LinkedList` class.

   1. `boolean add(E e)`
   2. `E get(int index)`
   3. `boolean contains(Object o)`
   4. `E remove()`

9. [8 points] Give the big-O time complexity for each of the following methods of the `ArrayList` class.

   1. `boolean add(E e)`
   2. `E get(int index)`
   3. `boolean contains(Object o)`
   4. `E remove(int index)`

10. [10 points] Fill in the blanks to complete the following implementation of the Binary Search algorithm on an array of Boolean values. Given an array that is sorted (false before true) in the half-open range `[begin,end)`, the function `find_first_true_sorted` should return the index of the first true element in the range `[begin,end)`, or if there is none, it should return `end`.

```
int find_first_true_sorted(boolean[] A, int begin, int end) {
    if (___(a)___) {
        return end;
    } else {
        int n = end - begin;
        int middle = ___(b)___;
        if (___(c)___) {
            return find_first_true_sorted(A, begin, ___(d)___);
        } else {
            return find_first_true_sorted(A, ___(e)___, end);
        }
    }
}
```

11. [10 points] The `fast_exp` function implements exponentiation, so `fast_exp`$(x, n) = x^n$ for $n \geq 0$. Is the following formula a loop invariant for the `while` loop? Explain in detail why or why not, analyzing the body of the loop, statement by statement.

$$y \cdot z^m = x^n$$

```
static int fast_exp(int x, int n) {
    if (n == 0) {
        return 1;
    }
    int y = 1, z = x, m = n;
    while (m != 1) {
        if (m % 2 == 0) {
            z = z * z;
            m = m / 2;
        } else {
            y = z * y;
            z = z * z;
            m = (m - 1) / 2;
        }
    }
    return z * y;
}
```