1. (20 points) What is the average and worst-case efficiencies of an insert function on the following ADTs?

   (a) Hash Table

   (b) Binary Search Tree (BST)

   (c) Heap

   (d) Linked List

   (e) Stack

2. (4 points) When using chaining for a hash table, what does the implementor need to be careful of?
3. (4 points) Construct a BST using input as follows: 21, 17, 69, 4, 75, 9, 2, 38, 73, 92.

4. (6 points) Explain how a breadth-first search works.

5. (4 points) Give two reasons why someone may wish to use parallel computing.
6. (4 points) What motivated the development of the BST?

7. (4 points) What motivated the development of the hash table?
8. (7 points) If you were to represent an assembly line with an ADT from our class, which would you use, and why?

9. (7 points) If you were to represent a phone book with an ADT from our class, which would you use, and why?
10. (6 points) How does a merge sort work?

11. (6 points) What are the three questions that can be used to ensure a recursive solution executes correctly?
12. (18 points) Draw a memory diagram for the following code.

```c
int *x = new int(5);
int *y = new int[3];
int **z = &y;
int *w = y + 1;
w[1] = 14;
*y = 88;
y = x;
x = *z;
x[1] = -2;
```

13. (3 points) What two statements need to be added to the above code during cleanup?
14. (4 points) What is an ADT?

15. (4 points) Which is faster, heap sort or insertion sort? Why?

16. (4 points) What is required of a good hash function?
17. (5 points) Describe a post-order traversal of a BST.

18. (3 points) How do you determine which test cases should be put into a test plan?