Answer the questions in the spaces provided on the question sheets. If you run out of room for an answer, continue on the back of the page.

Name: _____

1. (5 points) What is the definition of a recursive function?

2. (5 points) In your own words, what is the difference between a function and a method?

3. (10 points) Construct a *maximal* heap from the following array: $1, 2, 3, 4, 5, 6, 7$

0	1	2	3	4	5	6	7
X							
X							
X							
X							
X							
X							
X							
X							
X							
X							
X							
X							

4. (10 points) Rewrite the following recursive function to use iteration with an explicit stack.

```
int preorder(Tree t)
{
    std::cout << t.item << std::eol;
    if (t.left) {
        preorder(t.left);
    }
    if (t.right) {
        preorder(t.right);
    }
}
```

5. (10 points) Assume a linked list implementation identical to the 1st homework, or the **std::forward_list** of the standard template library. Write a recursive function that returns the sum of a linked list of integers given as its only argument.

```
int sum(LinkedList<int> list_to_sum)
{




}
```

6. (5 points) For each pair of operations, circle the operation that has the best performance. Assume that the linked list is identical to our Homework 1 implementation.

   (a)
   A. Get the i-th element of a vector.
   B. Get the i-th element of a linked list.
   C. These operations have the same performance.

   (b)
   A. `push_front` on a vector.
   B. `push_front` on a linked list.
   C. These operations have the same performance.

   (c)
   A. Binary search a vector.
   B. Binary search a linked list.
   C. These operations have the same performance.

   (d)
   A. `push_back` on a linked list.
   B. `push_front` on a linked list.
   C. These operations have the same performance.

   (e)
   A. `pop` on a stack.
   B. `push` on a stack.
   C. These operations have the same performance.

   (f)
   A. Preorder traversal of a binary tree.
   B. Postorder traversal of a binary tree.
   C. These operations have the same performance.

7. (5 points) Draw the graph represented by the following adjacency matrix.

$$\begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$

8. (5 points) In your own words, what information does the n-th power of the adjacency matrix give you?

9. (5 points) Convert the following infix expression into the equivalent prefix expression.

$$2 * 5 + 4 * (10/2) + 1$$

10. (5 points) Convert the following infix expression into the equivalent postfix expression.

$$3 * (6 + 1)/2 - 3$$

11. (10 points) Label the nodes of the following trees in the order they would be visited using the indicated algorithm.

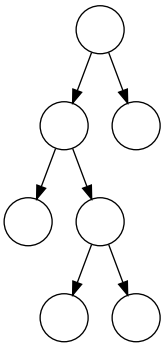


Figure 1: `preorder`

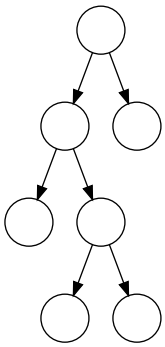


Figure 2: `inorder`

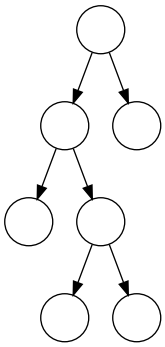
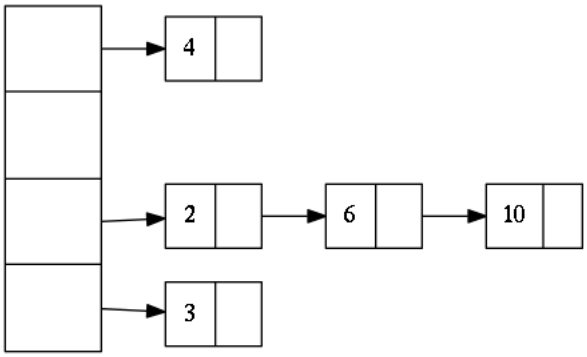


Figure 3: `postorder`

12. (10 points) Consider the following hashmap.



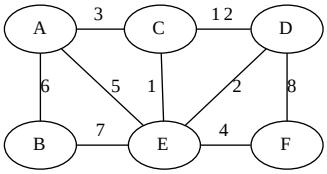
$N =$ the number of items
$M =$ the number of buckets
$k =$ the number of non-empty buckets

(a) Resize the following hashmap to a size of 8 and rehash the entries. Draw the result.

(b) If the load factor is defined as $\frac{N}{M}$, what is the load factor before and after the resize?

(c) What is the average lookup time before and after the resize?

The next several questions will use the following graph.



13. (5 points) Draw the adjacency list for the previous graph. (Ignore edge weights.)

14. (5 points) Write the adjacency matrix for the previous graph. (Ignore edge weights.)

15. (10 points) Perform Dijkstra's algorithm on the previous graph with vertex A as the source vertex.

Visiting	Distance from A to				
	B	C	D	E	F
A	1	3	∞	5	∞