CS 225 **Introduction to Computer Science** Test 2
Name: Maximum Points: 44

Important: This is an open book test. You can use any books, notes, or paper. *Do not log into the computer during the test.* Any calculations and rough work can be done on the back side of the test pages. If there is a syntax error in any program segment, just write it down and you will get full credit for the problem.

1. [6 pt] Write loop invariants for the following loop:

```
int frame, iters;
double e;
iters = 10;
for ( frame = 0, e = 0.0; frame < iters; e += exp ( -(frame++) * 0.33 ) );
```

   Remember that loop variants are assertions in plain English to be written as comments.

2. [4 pt] Calculate the big-O notation for the following:
   (a) $5n^{5/2} + n^{2/5}$
   (b) $6 \log_2 n + 9n$
   (c) $3n^4 + n \log_2 n$
   (d) $5n^2 + n^{3/2}$

3. [4 pt] Calculate the run-time complexity of the following program segment:

```
i = 1;
while ( i <= n )
{
    printf ( "%d\n", i );
    i++;
}
```

4. [6 pt] Imagine that we have two empty stacks of integers, $s1$ and $s2$. Draw a picture of each stack after the following operations. Identify the top of stack explicitly.

```
s1 = push ( s1, 3 );
s1 = push ( s1, 5 );
s2 = push ( s2, 7 );
s1 = push ( s1, 9 );
s1 = push ( s1, 11 );
s2 = push ( s2, 13 );
while ( ! empty_stack ( s1 ) )
{
    s2 = push ( s2, pop ( s1 ) );
}
```

5. [10 pt] What will be the value of queues $q1$, $q2$, and stack s after the following operations? Identify the top of stack and ends of queues explicitly.

```
int x, z;
s = create_stack ( s );
q1 = create_queue ( q1 );
q2 = create_queue ( q2 );
q1 = enqueue ( q1, 5 );
q1 = enqueue ( q1, 6 );
q1 = enqueue ( q1, 9 );
q1 = enqueue ( q1, 0 );
q1 = enqueue ( q1, 7 );
q1 = enqueue ( q1, 5 );
q1 = enqueue ( q1, 0 );
q1 = enqueue ( q1, 2 );
q1 = enqueue ( q1, 6 );

while ( ! empty_queue ( q1 ) )
{
    if ( ! ( x = dequeue ( q1, x ) ) )
    {
for ( z = 0; ! empty_stack ( s ); z += pop ( s ) );
q2 = enqueue ( q2, z );
    }
    else
        s = push ( s, x );
}
```

6. [6 pt] Using big-O notation, describe the efficiency of searching for an element, inserting an element, and deleting an element in a linked list when the list is represented using

   (a) an array

   (b) dynamically added nodes

7. [8 pt] Can we perform binary search on either of the list representations in the previous question? If yes, give the pseudocode algorithm for the same (for the case you can do it). If not, why not?