CS21 Lab 08, Swarthmore College, Spring 2011

Due at the beginning of class, Wednesday 30 March.

1. Suppose you use binary search to locate the value 24 in the list `ls` below.

```
def main():
    ls = [14, 19, 24, 31, 41, 57, 59, 63]
    result = binarySearch(24, ls)
    print result

main()
```

   (a) Show the stack frame diagram for this program up to the time immediately before `binarySearch` returns a value. As variables are updated, cross out the old value but do not erase the old values, so that we can see how `low`, `high`, and `mid` are updated.

   (b) What value is printed by the program?

2. What is the smallest n such that binary search might inspect 5 items in a list of size n? Give an exact answer, and explain your answer.

3. Consider the following function, `oneLoop`, which is similar to the inner loop from bubble sort:

```
def oneLoop(ls):
    for j in range(len(ls)-1):
        if ls[j] > ls[j+1]:
            tmp = ls[j+1]
            ls[j+1] = ls[j]
            ls[j] = tmp
```

   (a) Given the list `ls = [17, 4, 19, 3, 11, 8]`, what is `ls` after one call to `oneLoop(ls)`?

   (b) Given the list `ls = [17, 4, 19, 3, 11, 8]`, what is `ls` after two calls to `oneLoop(ls)`?

   (c) How many calls to `oneLoop(ls)` are needed before this list `ls` is sorted?

   (d) For any list `ls` of size n, how many calls to `oneLoop(ls)` are needed before `ls` is guaranteed to be sorted?

   (e) In (d), under what condition is the maximum number of calls needed? Explain your answer.

4. Rank the following running times from fastest to slowest: $O(n)$, $O(\log_2 n)$, $O(n^2)$, $O(n \log_2 n)$.