

# CSCI046 Midterm

**DUE: Friday, 13 March by noon**

## **Collaboration policy:**

You may not:

1. discuss the midterm exam with a human other than Mike; this includes:
  - (a) asking your friend for clarification about what a problem is asking
  - (b) asking your friend if they've completed the exam

You may:

1. take as much time as needed
2. use any written notes / electronic resources you would like
3. ask Mike to clarify questions

Name:

---

**Problem 1.** (2 pts) Complete each equation below by adding the symbol  $O$  if  $f = O(g)$ ,  $\Omega$  if  $f = \Omega(g)$ , or  $\Theta$  if  $f = \Theta(g)$ . The first row is completed for you as an example.

$f(n)$		$g(n)$
1	=	$O(n)$
$1.1^n$	=	$n^2$
$1/n$	=	1
$\log_4 n$	=	$\log_3 n$
$2^n$	=	$n^2$
$\log n$	=	$\sqrt{n}$
$\log(n^3)$	=	$\log n$
$\log(n!)$	=	$n \log n$
$n!$	=	$n^2$
$(\log n)^4$	=	$(\log n)^3$

**Problem 2.** (1 pt) Fill in the table with the appropriate runtimes in theta notation. You do not need to fill in the gray-ed out entries.

	run time			memory usage
	best case	worst case	average case	
bubble sort				
selection sort				
insertion sort				
merge sort				
quick sort				
tim sort				

**Problem 3.** (1 pt) What is a tight lower bound on the runtime (in omega notation) of the best possible comparison-based sorting algorithm?

**Problem 4.** (2 pts) Consider the following two functions:

---

```
1 def print_container_1(xs):
2     for i in range(len(xs)):
3         print('xs[i]=' ,xs[i])
4
5 def print_container_2(xs):
6     for x in xs:
7         print('x=' ,x)
```

---

Notice that both functions will work whether the **xs** parameter is a list or a deque. Whenever a function works for more than one type of input, we call it **polymorphic**.

1. Assume we have a **list** with **n** elements. What is the runtime of `print_container_1` on this list? (Use big-O notation.)
2. Assume we have a **list** with **n** elements. What is the runtime of `print_container_2` on this list? (Use big-O notation.)
3. Assume we have a **deque** with **n** elements. What is the runtime of `print_container_1` on this deque? (Use big-O notation.)
4. Assume we have a **deque** with **n** elements. What is the runtime of `print_container_2` on this deque? (Use big-O notation.)

**Problem 5.** We saw in class that binary search runs in logarithmic time. One possible idea for speeding up the computation is to perform trinary search (that is, split the list into three sections rather than two on each recursive call). The following code implements trinary search.

---

```
1 def trinary_search(xs, val):
2     left = 0
3     right = len(xs)
4     def go(left, right):
5         if right-left < 3:
6             return val in xs[left:right]
7         mid1 = left + (right-left)//3
8         mid2 = left + (right-left)//3*2
9         if val < xs[mid1]:
10            return go(left, mid1)
11        elif val < xs[mid2]:
12            return go(mid1, mid2)
13        else:
14            return go(mid2, right)
15    return go(left, right)
```

---

1. (2 pts) What is the recurrence relation describing the runtime of `trinary_search`?

2. (2 pts) Solve the recurrence relation you gave for part 1. (You can get full credit on this problem even if you gave the wrong answer on the previous problem.)