

CS 2210a Data Structures and Algorithms
Assignment 1 (20 marks)
Due September 27

Put your assignment in an envelope labelled with your name and course number and drop it in the CS2210 locker (locker 310 located on the third floor of the Middlesex College Building, besides room MC300) by midnight on the due date. You need to print an “Assignment ticket”; check

<http://www.csd.uwo.ca/FAQ/node47.html>, or
<http://www.csd.uwo.ca/forms/assnret.mysql.html>

for information on how to print the ticket. You must also fill out an assignment submission form:
<http://www.csd.uwo.ca/courses/CS2210a/submForm.html>.

Put the assignment ticket and submission form in the envelope along with your assignment.

Remember that when you are asked to find the time complexity of an algorithm you are required to give a big-Oh characterisation in terms of n of the running time of the algorithm. You might find the following fact useful: $\sum_{i=1}^n i = \frac{n(n+1)}{2}$.

1. (3 marks) Use the definition of “big Oh” to prove that $3n^2 + 4n$ is $O(n^2)$.
2. (3 marks) Use the definition of “big Oh” to prove that n is not $O(\sqrt{n})$.
3. (3 marks) Let $f(n)$ and $g(n)$ be non-negative functions. Use the definition of “big Oh” to prove that $k \times f(n) - k' \times g(n)$ is $O(f(n))$, where k and k' are positive constants.
4. Let A be an array storing n different integer values. Given an integer value k , we say that A is *k-flat* if for each value $A[i]$ there is another value $A[j]$, $j \neq i$ such that $A[i] + A[j] = k$. For example, for $k = 11$ the following array is *k-flat*, as $A[0] + A[2] = A[1] + A[3] = A[4] + A[5] = 11$.

9	7	2	4	8	3
0	1	2	3	4	5

However, the same array is not *k-flat* for $k = 10$, as there is no value in A that added to $A[0]$ gives 10.

- i.* (4 marks) Write pseudocode for a simple algorithm that receives as input an array A as above, the size n of the array, and a value k , and it returns `true` if A is *k-flat* and it returns `false` otherwise.
 - ii.* (4 marks) Explain what the worst case for the algorithm is and compute the time complexity of the algorithm in the worst case. You must explain how you computed the time complexity.
5. (3 marks) Consider the following algorithm:

Algorithm Foo (n)

```
 $x \leftarrow 0$   
for  $i \leftarrow 0$  to  $n \times n$  do  
    for  $j \leftarrow 0$  to  $i$  do  
         $x \leftarrow x + i$   
  
return  $x$ 
```

Compute the time complexity of this algorithm in the worst case. Explain how you computed the complexity.

6. (2 marks) **Optional question.** Download from the course's website the java class `Search.java`, which contains implementations of 3 different algorithms for solving the search problem:

- `LinearSearch`, of time complexity $O(n)$.
- `QuadraticSeach`, of time complexity $O(n^2)$.
- `FactorialSearch`, of time complexity $O(n!)$.

Modify the `main` method so that it computes the worst case running times of the above algorithms for the following input sizes:

- `FactorialSearch`, for input sizes $n = 5, 8, 9, 10, 11$.
- `QuadraticSeach`, for input sizes $n = 5, 10, 100, 1000, 2000$.
- `LinearSearch` for, input sizes $n = 5, 10, 100, 1000, 2000, 10000$.

Print a table indicating the running times of the algorithms for the above input sizes. You do not need to include the code for the `Search` class.