CSCI 314 – Fall 2012 Handout: asgn01
Algorithms
Michael H. Goldwasser
Saint Louis University Wednesday, 29 August 2012

Homework #1:	Introduction
Due Date:	Friday, 7 September 2012

Guidelines

Please make sure you adhere to the policies on collaboration and academic honesty as outlined in the syllabus.

Reading

Chapters 1, 2, 4.1, 10 of CLRS, as well as the lecture notes regarding the maximum subarray problem.

Problems

There are four required problems worth 25 points each, and one extra credit challenge worth 10 points.

Problem A (25 points)

“Work entirely on your own.”

In class, we discussed five algorithms for the maximum subarray problem, based on Chapter 8 of Jon Bentley’s *Programming Pearls*. In this problem, you are to experiment with the efficiency of the five algorithms. You are to run each of the algorithms on random data sets of length n for varying values of n. You are to complete as much of the following table as possible. For each column, fill in as many times as you can reasonably gather, stopping when the running times become “unreasonable” (e.g., longer than 5–10 minutes?)

	Algorithm									
	1		2		2b		3		4	
n	time	$\frac{\text{time}}{n^3}$	time	$\frac{\text{time}}{n^2}$	time	$\frac{\text{time}}{n^2}$	time	$\frac{\text{time}}{n \log n}$	time	$\frac{\text{time}}{n}$
...										
2^{10}										
2^{12}										
2^{14}										
2^{16}										
2^{18}										
2^{20}										
2^{22}										
...										

You may use the programming language of your choice, but we note that Bentley provides C implementations of all five algorithms, and we have provided Python implementations of all five (links can be found in the lecture notes). Running times will certainly depend upon the choice of language and the computing environment; just try to do all experiments under similar conditions.

Problem B (25 points)

“Work entirely on your own.”

CLRS Exercise 10.3-4 (page 245).

Problem C (25 points)

"You may discuss ideas with other students."

Assume that we have a sequence of n numbers containing all the integers from 0 to n except one, in arbitrary order. The goal is to efficiently determine which number is missing. However, the numbers are represented in binary. For example, the input for $n = 5$ might be modeled as:

1	0	1
0	1	1
0	1	0
0	0	0
0	0	1

By the way, in this example, the missing number is 4.

In solving the problem, you will be charged for each **bit** of the input that is examined. Therefore, at the beginning of the process for a given n, the unknown input might be viewed as follows:

?	?	?
?	?	?
?	?	?
?	?	?
?	?	?

Note that the input to the algorithm consists of an array of n numbers, each of which is comprised of $\lfloor \lg n \rfloor + 1$ bits.

Describe an algorithm that finds the missing number while fetching $O(n)$ bits. Make sure to justify its correctness and analysis.

Problem D (25 points)

"You may discuss ideas with other students."

CLRS Exercise 10.2-8 (page 241).

Problem E **(EXTRA CREDIT – 10 points)**

"You may discuss ideas with other students."

CLRS Problem 4-5 (pages 109–110).