CSE 101 Homework 1

Due October 14, 2004

1. Consider the following algorithm which, given a natural number \( x \geq 2 \), computes the largest prime number that is less than or equal to \( x \). The algorithm is called the “sieve of Eratosthenes.”

Precondition: \( x \) is a natural number and \( x \geq 2 \)
Postcondition: algorithm PRIME outputs largest prime number \( y \) such that \( y \leq x \)

```plaintext
integer PRIME (integer x)
   Allocate array A[1..x] of integers (initially all 0)
   A[1] := 1
   i := 2
   While i <= x do
      FIND-MULTIPLES (A, i, x)
      i := i + 1
   End While
   For i = x downto 2 do
      If A[i] = 0 then
         Return i
      End If
   End For
End
```

```plaintext
FIND-MULTIPLES (A, i, x)
   j := 2*i
   While j <= x do
      A[j] := 1
      j := j + i
   End While
End
```

(a) Say that a number \( k \) is a nontrivial multiple of \( i \) if \( k \neq i \), but \( i | k \). Assume that FIND-MULTIPLES(A, i, x) is correct; that is, if \( k \) is a nontrivial multiple of \( i \), it sets \( A[k] \) to 1. Otherwise, it doesn’t change \( A[k] \). Let \( A^0 \) be the set \( \{ k : 1 \leq k \leq x, A[k] = 0 \} \). State and prove a loop invariant for the While loop in PRIME that implies that when the loop terminates, \( A^0 \) is exactly the set of primes less than or equal to \( x \) (hint: let \( A_t \)
denote the array $A$ after $t$ iterations of the loop, and let $i_t$ denote the value of $i$ after $t$
iterations).

(b) Let $n$ be the number of bits in $x$. What is the worst-case complexity of PRIME, using
$\Theta$-notation, in terms of $n$? Assume that accessing $A[k]$ for $1 \leq k \leq x$ takes time $\Theta(n)$.

2. Recall from lecture that the Fibonacci sequence is defined as follows: $F_n = F_{n-1} + F_{n-2}$,
where $F_1 = 1$ and $F_2 = 1$. One very natural algorithm for computing $F_n$ is the following:

```c
int Fib (int n)
    If n = 1
        (*) Return 1
    Else if n = 2
        (*) Return 1
    Else
        Return Fib(n-1) + Fib(n-2)
End
```

(a) Let’s measure the complexity of this algorithm in terms of the number of times the (*)
lines are executed in all recursive calls. What is the complexity of computing $F_n$?

(b) Give an iterative algorithm (one that uses loops instead of recursion) for computing $F_n$
that runs in time $O(n^2)$.

3. An $m \times n$ Young Tableau is an $m \times n$ matrix of natural numbers such that each row is sorted
in ascending order and each column is sorted in ascending order. The value $\infty$ can appear in
a tableau—this is treated as an empty slot in the tableau. There can be many empty slots.

(a) Show that an $m \times n$ Young Tableau $Y$ is empty if $Y[1, 1] = \infty$ and is full if $Y[m, n] < \infty$.

(b) Given any set of natural numbers $A = \{a_1, \ldots, a_k\}$, where $k \leq mn$, is there a valid $m \times n$
Young Tableau that contains exactly $A$ (plus empty slots if $k < mn$)? Explain.

(c) Give an $O(m + n)$ algorithm for removing the minimum element of a Young Tableau $Y$
(hint: think Delete-Min for heaps).

(d) Give an $O(m + n)$ algorithm for inserting an element into a non-full Young Tableau $Y$.

(e) Give an $O(m + n)$ algorithm for determining whether a certain number $x$ appears in a
given Young Tableau $Y$.

4. Prove that any graph with $n$ vertices and $m \geq n$ edges contains a cycle (Hint: use the lemmas
from last lecture).