CSE 417
Algorithms & Computational Complexity
*** DRAFT *** Assignment #1
Due: Friday, 1/14/22

Note On Drafts: This version is *not* final, and I am likely to add/reword/renumber problems, but I am very unlikely to delete a problem outright after posting it, even in draft form, so it is safe to start working on them now. Just re-read the assignment carefully after it is finalized to be sure there are no surprises...

Instructions:

- **Handwriting:** Typed solutions are *strongly* encouraged; (a) handwriting is hard to read/grade, and (b) your ability to easily revise will lead you to clearer writing, clearer thinking, better scores, and perhaps whiter teeth, less dandruff, more twitter followers... To type the math:
  - The equation editors in Word & its competitors are OK (tho perhaps a bit cumbersome).
  - Plain text (suffix .txt) is OK, provided you're clear about subscripts and other math notation. E.g., use `x_{i,j}^{2}` for "the square of x sub i,j".
  - LaTeX is excellent for math. Like all good tools, it has a learning curve, but the basics are easy. E.g., `x_{i,j}^{2}` $\rightarrow x_{i,j}^{2}$ and `{{n}\choose{k}}` $\rightarrow \binom{n}{k}$. Here is a brief intro. It has simple, free installers for Windows, Mac and Unix.
  - Even typed prose with handwritten math is welcome, esp. if your handwriting is as bad a mine...
- **Mechanics:** Assignments must turned in *electronically*, via Gradescope; Details coming. These details are important; read carefully.
  - Each problem will be turned in *separately* to a problem-specific dropbox on Gradescope.
  - Make sure your name, student number and UWNetID are clearly at the front of each.
  - If your solution is partially or (horrors!) completely handwritten: use dark ink, or firm, dark pencil. Scan it to a file. Many campus labs (e.g., these) have scanners. In a pinch, a carefully lit, focused shot from a good camera-phone may suffice. In any case, TEST this before the last minute...
  - Turn in a .pdf file (or plain .txt, if that's your formatting choice). (***NO** word processor source files (.doc, .docx, .pages, .tex, .xps, ...); we've had myriad compatibility issues.)*
  - In all cases, make sure pages are in order, legible, not rotated, not upside down, ...

Important Note:

For all homework assignments this quarter, it is a violation of my academic integrity policy for you to search for, read, or use solutions to these or similar problems written by others. You may discuss these problems with other students in this class, but you must *write your solutions on your own*.

Problems:

1. Briefly explain (if true) or give a counterexample (if false): In every instance of the Stable Matching Problem, there is a stable matching containing a pair (m, w) such that m is w's first choice and w is m's first choice.
2. Briefly explain (if true) or give a counterexample (if false): If, in a stable matching instance, there is a pair \$(m,w)\$ such that \$m\$ is \$w\$'s first choice and \$w\$ is \$m\$'s first choice, then \$m\$ and \$w\$ are matched to each other in every stable solution.
3. The Team USA Winter Olympic 5-Person Snowball Throwing Trials are approaching. There are n coaches available; each will coach a single 5-Person Snowball Throwing Team ("5PSTT" for short). $5n$ athletes are eagerly competing for these highly coveted 5PSTT slots. After practices, trials and training, each athlete ranks the n coaches, and each coach ranks all $5n$ athletes. A "stable team assignment" assigns 5 athletes to each coach so that no coach c is assigned and athlete a who would prefer a different coach c' and simultaneously c' is assigned an athlete a' who prefers coach c. This is similar, but not identical, to the Stable Matching Problem

(SMP). Explain how you could use, *without modification*, a subroutine that solves SMP to solve the Stable 5PSTT Problem (S5PSTTP). I.e., explain how to transform an S5PSTTP instance x into an SMP instance $f(x)$ in such a way that any stable solution to the SMP instance $f(x)$ can be easily converted into a stable solution to the S5PSTTP instance x. I'm looking for about 3 clear, English language, paragraphs explaining (1) how to convert x to $f(x)$ (2) how to recover a solution to the S5PSTTP instance x from a Stable Matching found in $f(x)$, and (3) explaining why that result is actually a "stable team assighnment". I expect that your solution would require $O(n)$ time, *excluding* the time taken by the SMP subroutine; *briefly* justify this (if true), or explain why your method may be slower.

4. Assuming that your computer can perform 10 billion operations per second, what is the largest value of n such that you can complete the following number of operation in one hour?
   (a) n^2
   (b) n^3
   (c) $100n^2$
   (d) $n \log_2 n$
   (e) 2^n
   (f) 2^{2^n}

5. Arrange the following six function in order of non-decreasing growth rate; i.e., f_i may precede f_j only if $f_i = O(f_j)$.
   (a) $f_1(n) = n^{2.5}$
   (b) $f_2(n) = \sqrt{2n}$
   (c) $f_3(n) = n + 10$
   (d) $f_4(n) = 10^n$
   (e) $f_5(n) = 100^n$
   (f) $f_6(n) = n^2 \log_2 n$

6. Suppose functions $f(n) > 0$ and $g(n) > 0$ satisfy $f(n) = O(g(n))$. Prove or give a counterexample to each of the following.
   (a) $\log_2 f(n) = O(\log_2 g(n))$. To avoid an obscure special case, assume that $g(n) > 2$ for all n.
   (b) $2^{f(n)} = O(2^{g(n)})$
   (c) $(f(n))^2 = O((g(n))^2)$

1. Briefly explain (if true) or give a counterexample (if false): In every instance of the Stable Matching Problem, there is a stable matching containing a pair (m, w) such that m is w's first choice and w is m's first choice.

False.

Given (m, m'), (w, w')

in each stable Matching Problem. men choose first

∴ (m, w), (m', w')

choice of m matters, so w can be m's first choice,

but not guarantee when it comes to w's Choice.

2. Briefly explain (if true) or give a counterexample (if false): If, in a stable matching instance, there is a pair \$(m,w)\$ such that \$m\$ is \$w\$'s first choice and \$w\$ is \$m\$'s first choice, then \$m\$ and \$w\$ are matched to each other in every stable solution.

True.

given pairs (m, m'). (w, w')

if (m, w) is not matched in every solution

then it will be pairs (m, w'), (m', w) in it.

However, m likes w at first place, which will lead to a unstable instance. This is not consistent with the given "stable" attribute.

4. Assuming that your computer can perform 10 billion operations per second, what is the largest value of n such that you can complete the following number of operation in one hour?

   (a) n^2
   (b) n^3
   (c) $100n^2$
   (d) $n \log_2 n$
   (e) 2^n
   (f) 2^{2^n}

$10^{10} \times 60 \times 60 = 3.6 \times 10^{13}$ operations/hour

a) $n^2 = 3.6 \times 10^{13}$

$n = 6 \times 10^6$

b) $n^3 = 3.6 \times 10^{13}$

$n = 3.3019 \times 10^4$

c) $100n^2 = 3.6 \times 10^{13}$

$n^2 = 3.6 \times 10^{11}$

$n = 6 \times 10^5$

d) $n \log_2 n = 3.6 \times 10^{13}$

$n = 9.06316 \times 10^{11}$

e) $2^n = 3.6 \times 10^{13}$

$n = 45$

f) $2^{2^n} = 3.6 \times 10^{13}$

$n = 5$

5. Arrange the following six function in order of non-decreasing growth rate; i.e., f_i may precede f_j only if $f_i = O(f_j)$.

(a) $f_1(n) = n^{2.5}$

(b) $f_2(n) = \sqrt{2n}$

(c) $f_3(n) = n + 10$

(d) $f_4(n) = 10^n$

(e) $f_5(n) = 100^n$

(f) $f_6(n) = n^2 \log_2 n$

$\because 10 < 100$

$\therefore f_4 < f_5$

$\because 0.5 < 1 < 2.5$

$\therefore f_2 < f_3 < f_1$

Let $n = 1$

$\because 1^{2.5} > \log_2$

$\therefore f_6 < f_1$

$\therefore f_2 < f_3 < f_6 < f_1 < f_4 < f_5$

6. Suppose functions $f(n) > 0$ and $g(n) > 0$ satisfy $f(n) = O(g(n))$. Prove or give a counterexample to each of the following.

(a) $\log_2 f(n) = O(\log_2 g(n))$. To avoid an obscure special case, assume that $g(n) > 2$ for all n.

(b) $2^{f(n)} = O(2^{g(n)})$

(c) $(f(n))^2 = O((g(n))^2)$

a) $\log_2 f(n) = O(\log_2 g(n))$

$\log_2 f(n) \leq C \cdot \log_2 g(n)$

$\because$ $\log_2 f(n) \leq C \cdot \log_2 g(n)$ is valid

$\therefore$ it is true when $C > 0$ and $g(n) > 2$.

b) $2^{f(n)} = O(2^{g(n)})$

$2^{f(n)} \leq C \cdot 2^{g(n)}$

Let $f(n) = 2n$ and $g(n) = n$ $\rightarrow$ $2n < C \cdot n$ when $C > 2$

$2^{2n} \leq C \cdot 2^n \Rightarrow 4^n \leq C \cdot 2^n$

$\because$ the function will not be valid when n is increasing towards infinity ∞.

$\therefore$ it is false.

c) $(f(n))^2 = O((g(n))^2)$

$(f(n))^2 \leq c \cdot (g(n))^2$

$\because (f(n))^2 \leq c \cdot (g(n))^2$ is valid

$\therefore$ it is true when $c > 0$