# CS 180 Summer 25 – Homework 1

Due Saturday, July 5, 11:59pm

- Please write your student ID and the names of anyone you collaborated with on the first page of your submission. Do not include your name on your assignment as the homework will be blind graded.
- You may use any theorem proven in class or in the textbook without proof. When referring to these theorems, write "as was proven in class" or "as was proven in the textbook" accordingly.
- **If you are asked to write an algorithm, you must prove its correctness.** If you are asked for a time complexity analysis, then you should give a reasonable big-O notation upper bound.
- The starred problems (Problems 4, 5, and 6) will be graded by correctness.
- All other problems (Problems 1, 2, 3, and 7) will be graded by completeness. You will receive full credit on these problems as long as a good-faith effort was made to solve them. This includes showing your work and providing proofs of your statements.

1. **Proof Techniques**

   (a) Prove by induction on $n$ that for all integers $n \geq 1$,

   $$1 + 4 + 7 + \ldots + (3n - 2) = \frac{n(3n-1)}{2}.$$

   (b) Let $x$ and $y$ be positive real numbers. Prove by contradiction: If $x^2 - y^2 = 1$, then $x$ or $y$ (or both) are not integers.

2. **Order Notation**

   Indicate the relationships between each pair of functions $(A, B)$ listed below by writing a "yes" or a "no" in each empty box of the table below. For example, if $A \in o(B)$, then you should write a "yes" in the corresponding row of the first empty column. If the base of a logarithm is not specified, you should assume it is base 2. For this problem, you do not have to show your work or justify your answer.

| A | B | A $\in o$(B) | A $\in O$(B) | A $\in \theta$(B) | A $\in \Omega$(B) | A $\in \omega$(B) |
|---|---|---|---|---|---|---|
| $\log_2(n)$ | $\log_5(n)$ | | | | | |
| $\log\log(n)$ | $\sqrt{\log(n)}$ | | | | | |
| $n^{4/3}\log(n)$ | $n^{\log(n)}$ | | | | | |
| $2^{\log^6(n)}$ | $n^6$ | | | | | |
| $n^3 2^n$ | $3^n$ | | | | | |

3. **Runtime Analysis**

   For each pseudocode snippet below, give the asymptotic running time in $\Theta$ notation and explain how you derived it. Assume that basic arithmetic operations $(+, -, \times, /)$ are constant time.

   (a)

   **for** $i = 1$ *to* $n$ **do**
   $\quad j = 0;$
   $\quad$ **while** $j \leq i$ **do**
   $\quad\quad j = j + 2;$
   $\quad$ **end**
   **end**

   (b)

   $i = 2;$
   **while** $i \leq n$ **do**
   $\quad i = i^2;$
   **end**

4. ***Smallest Element with Minimum Frequency**
   *(Graded by Correctness)*

   Let $A$ be an array of size $n$ which contains only positive integers. Give an algorithm which takes as input such an array $A$ and finds the SMALLEST element with MINIMUM frequency. Additionally, provide a time complexity analysis of your algorithm in terms of $n$.

   **Example**

   - **Input**: $n = 5, A = [2, 2, 7, 50, 4]$
   - **Output**: 4
     (All values have frequency 1 except the value 2. Thus, 4 is the smallest element with minimum frequency.)

5. ***Stable Matching with Unequal Numbers and Multiple Partners (Problem 1.4 in the Textbook)**
*(Graded by Correctness)*

Gale and Shapley published their paper on the Stable Matching Problem in 1962; but a version of their algorithm had already been in use for ten years by the National Resident Matching Program, for the problem of assigning medical residents to hospitals.

Basically, the situation was the following. There were m hospitals, each with a certain number of available positions for hiring residents. There were n medical students graduating in a given year, each interested in joining one of the hospitals. Each hospital had a ranking of the students in order of preference, and each student had a ranking of the hospitals in order of preference. We will assume that there were more students graduating than there were slots available in the m hospitals.

The interest, naturally, was in finding a way of assigning each student to at most one hospital, in such a way that all available positions in all hospitals were filled. (Since we are assuming a surplus of students, there would be some students who do not get assigned to any hospital.)

We say that an assignment of students to hospitals is stable if neither of the following situations arises.

- First type of instability: There are students $s$ and $s'$ and a hospital $h$ such that
  - $s$ is assigned to $h$, and
  - $s'$ is assigned to no hospital, and
  - $h$ prefers $s'$ to $s$.
- Second type of instability: There are students $s$ and $s'$, and hospitals $h$ and $h'$ such that
  - $s$ is assigned to $h$, and
  - $s'$ is assigned to $h'$, and
  - $h$ prefers $s'$ to $s$, and
  - $s'$ prefers $h$ to $h'$.

So we basically have the Stable Matching Problem, except that (i) hospitals generally want more than one resident, and (ii) there is a surplus of medical students.

Show that there is always a stable assignment of students to hospitals, and give an algorithm to find one. Additionally, provide a time complexity analysis of your algorithm.

6. ***Peripatetic Shipping Lines (Problem 1.6 in the Textbook)**
*(Graded by Correctness)*

Peripatetic Shipping Lines, Inc., is a shipping company that owns n ships and provides service to n ports. Each of its ships has a schedule that says, for each day of the month, which of the ports it's currently visiting, or whether it's out at sea. (You can assume the "month" here has m days, for some m > n.) Each ship visits each port for exactly one day during the month. For safety reasons, PSL Inc. has the following strict requirement:

$(*)$ No two ships can be in the same port on the same day.

The company wants to perform maintenance on all the ships this month, via the following scheme. They want to truncate each ship's schedule: for each ship $S_i$, there will be some day when it arrives in its scheduled port and simply remains there for the rest of the month (for maintenance). This means that $S_i$ will not visit the remaining ports on its schedule (if any) that month, but this is okay. So the truncation of $S_i$'s schedule will simply consist of its original schedule up to a certain specified day on which it is in a port $P$; the remainder of the truncated schedule simply has it remain in port $P$.

Now the company's question to you is the following: Given the schedule for each ship, find a truncation of each so that condition $(*)$ continues to hold: no two ships are ever in the same port on the same day.

Show that such a set of truncations can always be found, and give an algorithm to find them. Additionally, provide a time complexity analysis of your algorithm.

**Example.** Suppose we have two ships and two ports, and the "month" has four days. Suppose the first ship's schedule is

port $P_1$; at sea; port $P_2$; at sea

and the second ship's schedule is

at sea; port $P_1$; at sea; port $P_2$

Then the (only) way to choose truncations would be to have the first ship remain in port $P_2$ starting on day 3, and have the second ship remain in port $P_1$ starting on day 2.

**Hint.** Try to relate this problem to the stable matching problem.

7. **Truthfulness in the Stable Matching Problem (Problem 1.8 in the Textbook)**

For this problem, we will explore the issue of truthfulness in the Stable Matching Problem and specifically in the Gale-Shapley algorithm. The basic question is: Can a man or a woman end up better off by lying about his or her preferences? More concretely, we suppose each participant has a true preference order. Now consider a woman w. Suppose w prefers man m to m', but both m and m' are low on her list of preferences. Can it be the case that by switching the order of m and m' on her list of preferences (i.e., by falsely claiming that she prefers m' to m) and running the algorithm with this false preference list, w will end up with a man m" that she truly prefers to both m and m'? (We can ask the same question for men, but will focus on the case of women for purposes of this question.)

Resolve this question by doing one of the following two things:

(i.) Give a proof that, for any set of preference lists, switching the order of a pair on the list cannot improve a woman's partner in the Gale-Shapley algorithm; or

(ii.) Give an example of a set of preference lists for which there is a switch that would improve the partner of a woman who switched preferences.