DISCRETE MATH 37181 WORKSHEET 6

MURRAY ELDER

ABSTRACT. Complete these problems in small groups at the whiteboards. No quiz this week: if you finish the worksheet early, get started on the team assignment. (Your team could be other students in your workshop, or from a different workshop if you prefer.)

Big-O form	Name
$O(1)$	constant
$O(\log_2 n)$	logarithmic
$O(n)$	linear
$O(n \log_2 n)$	$n \log n$ (sometimes called quasilinear)
$O(n^2)$	quadratic
$O(n^3)$	cubic
$O(n^p), p \in \mathbb{N}$	polynomial
$O(c^n), c > 1$	exponential
$O(n!)$	factorial

TABLE 1. Some standard functions for comparison.

1. For each of the following functions, guess a Big-O form from one of those in Table 1, then prove your guess.

   (a) $f(n) = 4n + 7$

   (b) $f(n) = 3 + \sin(n)$

   (c) $f(n) = 5n^2 + 3n \log_2 n$

   (d) $f(n) = 2 + 4 + 6 + \cdots + 2n$

   (e) $f(n) = n + \frac{1}{n}$

   (f) $f(n) = \log_4 n + 7$

   (g) $f(n) = \frac{1}{1.2} + \frac{1}{2.3} + \frac{1}{3.4} + \cdots + \frac{1}{n(n+1)}$

   (h) $f(n) = 1 + 2^3 + 3^3 + \cdots + n^3$

2. Let $f, g : \mathbb{N} \to \mathbb{N}$ defined by $f(n) = \begin{cases} n & \text{for } n \text{ odd} \\ 1 & \text{for } n \text{ even} \end{cases}$ and $g(n) = \begin{cases} 1 & \text{for } n \text{ odd} \\ n & \text{for } n \text{ even} \end{cases}$

   (a) Draw plots of $f(n)$ and $g(n)$.

   (b) Is $f \in O(g)$ or $g \in O(f)$?

3. Prove that *domination* is reflexive and transitive[1]. Is it symmetric and/or antisymmetric?

Date: Week 6.

[1]i.e. for all $f, g, h : \mathbb{N}_+ \to \mathbb{R}$, $f \in O(f)$, and if $f \in O(g)$ and $g \in O(h)$ then $f \in O(h)$.

4. Here is a pseudocode program. (In this code arrays are numbered starting at 1).

```
procedure Foo (n int; array=[a_1, a_2, ..., a_n] )
  begin
  num := array[1]
  for i := 2 to n do
        if array[i]> num
            num := array[i]
  print num
  end
```

   Work out what it does, try it out on input $(5; [3, 1, 4, 2, 0])$, then compute a Big-O complexity for how many steps it takes in worst-case. A *step* can be considered as how many times the comparison

   `a_i > num`

   is performed.

5. Here is some more pseudocode.

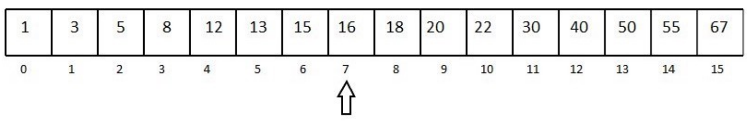
```
procedure Bar (n int; array=[a_1,...,a_n] array of integers)
   begin
   for i := 2 to n do
       for j := 1 to n-i-1 do
            if array[j] > array[j+1]
                  temp := array[j]
                  array[j] := array[j+1]
                  array[j+1] := temp
   print array
   end
```

   Work out what it does, try it out on input $(5; [3, 1, 4, 2, 0])$, then compute a Big-O complexity for how many steps it takes in worst-case. A *step* can be considered as how many times the comparison

   `array[j] > array[j+1]`

   is performed. [2]

6. Consider the following problem: input is a sorted array of distinct integers, eg:



1	3	5	8	12	13	15	16	18	20	22	30	40	50	55	67
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

   Devise an efficient algorithm to locate a particular entry of the array, say the number 50. How many steps (Big O) roughly does your algorithm need? Hint: don't start at the start of the array and move left!
   [3]

   Also, will your algorithm be able to return No if the entry does not appear?

[2]see `http://www.geeksforgeeks.org/bubble-sort/` if you prefer to read real C, Java or Python code.
[3]Hint here: `https://hackernoon.com/what-does-the-time-complexity-o-log-n-actually-mean-45f94bb5bfbf`

Recall:

Axiom 1 (Pigeonhole principle)**.** *If m pigeons occupy n pigeonholes and $m > n$ then some pigeonhole has at least two pigeons in it.*

To give a proof using it, you need to specify what are the m pigeons, what are the n pigeonholes, and some rule for placing pigeon x in pigeonhole y.

Axiom 2 (Generalised Pigeonhole principle)**.** *If m pigeons occupy n pigeonholes and $m > kn$ then some pigeonhole has more than k pigeons in it.*

To give a proof using it, you need to specify what are the m pigeons, the n pigeonholes, some rule for placing pigeon x in pigeonhole y, and what is the k.

7. Use PHP to show that if $S \subseteq \mathbb{N}_+$ and
   (a) $|S| \geqslant 3$ then S contains two distinct elements x, y such that $x + y$ is even.
   (b) $|S| > 6$ then S contains three distinct elements x, y, z such that $x + y + z$ is a multiple of 3.

8. There are 51 students taking a course, and the available grades are Z, P, C, D, HD. Show that at least 11 students will get the same grade.

9. Let p be an odd number and let S be any subset of $\mathbb{Z}_{p+1} = \{0, 1, \ldots, p\}$ which contains at least $2 + \frac{p-1}{2}$ elements. Show that there are at least two elements of S with sum equal to p.

10. Let Δ be an equilateral triangle with side length 1. Show that if we select 10 points in the interior of Δ then at least two of them lie distance $< \frac{1}{3}$ of each other. [4]

11. A factory produces robots. It needs to fill an order of 44 robots in 30 days, and in order to maintain the machinery it must produce at least one robot a day. Prove that there must be a string of some number of consecutive days where the factory produces exactly 15 robots. [5]

12. In computer science it is useful to *compress* strings. For example in a text file we might want to replace all strings *the* by κ and *murray* by $\delta\#$. For this, we assume we have a fixed finite *alphabet* of symbols available, say $a, b, c, \ldots, z, A, B, C, \ldots, Z, 0, 1, 2, 3, \ldots, 9$, plus a *space* symbol plus some finite list of special symbols $\kappa, \delta, \#, \ldots$.
    A *universal lossless compression algorithm* is an algorithm that rewrites *every* string as something shorter. Show that such an algorithm cannot exist (use PHP).

Remember: partial solutions are always available to check, at the end of the pdf document on UTS online.

[4]hint: divide the triangle up into 9 smaller triangles

[5]hint: Let x_i be the number of robots produced between day 1 and day i. Let $y_i = x_i + 15$. Use PHP to show that some $x_i = y_j$.

Brief solutions:

1. (a) Linear. $m = 5, k = 7$ then for all $n \geqslant 7$ we have $5n = 4n + n \geqslant 4n + 7$ so $mg(n) \geqslant f(n)$ holds.

   (b) Constant. $g(n) = 1$, since $\sin(n)$ bounces around between -1 and 1. Proof: for $m = 4, k = 1$ we have
   $$f(n) = 3 + \sin(n) \leqslant 3 + 1 = 4 = 4.g(n)$$
   for all $n \geqslant 1$.

   (c) Quadratic. $\log_2 n \leqslant n$ for $n \geqslant 1$, so for $k = 1$ and $m = 8$ we have
   $$f(n) = 5n^2 + 3\log_2 n \leqslant 5n^2 + (3n).n = 8n^2 = 8g(n)$$
   for all $n \geqslant 1$.

   (d) Induction we know that this is $2\sum_{i=1}^{n} i = n(n+1)$ so: quadratic. $f(n) = n(n+1) = n^2 + n$, $g(n) = n^2$, then $m = 2, k = 1$ we have
   $$f(n) = n^2 + n \leqslant n^2 + n.n = 2n^2 = 2g(n)$$
   for all $n \geqslant 1$.

   (e) Linear. $m = 2, k = 1$. For $n \geqslant 1$ we have
   $$f(n) = n + \frac{1}{n} \leqslant n + 1$$
   since $\frac{1}{n} \leqslant 1$
   $$\leqslant n + n = 2n = 2g(n).$$

   (f) Logarithmic.
   $$f(n) = \log_4 n + 7 = \frac{\log_2 n}{\log_2 4} + 7$$
   using $\log_b y = \frac{\log_a y}{\log_a b}$
   $$= \frac{\log_2 n}{2} + 7 \leqslant \frac{\log_2 n}{2} + \frac{\log_2 n}{2}$$
   since $\log_2 n \geqslant 14$ for n sufficiently big, so $k = 2^{15}$ would make this true, and $m = 1$.

   (g) Induction we know this is $\frac{n}{n+1}$ which is always less than 1, so $g(n) = 1$, Constant, $m = k = 1$.

   (h) Induction we proved a formula for this in Quiz 4:

   Lemma 3. *For all* $n \in \mathbb{N}_+$
   $$1^3 + 2^3 + \cdots + n^3 = \frac{n^2(n+1)^2}{4}$$
   So $O(n^4)$.

2. (b) Neither is true.

3. Let $f, g, h : \mathbb{N}_+ \to \mathbb{R}$ be three functions from positive integers to real numbers.

   Then $f \in O(f)$ since there exists $m = 1, k = 1$ so that $f(n) \leqslant 1 \cdot f(n)$ for all $n \geqslant 1$, so domination is reflexive.

   By definition there exists $m_1, m_2 \in \mathbb{R}_+$ and $k_1, k_2 \in \mathbb{N}_+$ with $|f(n)| \leqslant m_1|g(n)|$ for $n \geqslant k_1$ and $|g(n)| \leqslant m_2|h(n)|$ for $n \geqslant k_2$

   so letting $m = m_1m_2$ and $k = \max\{k_1, k_2\}$ we have for all $n \geqslant k$
$$|f(n)| \leqslant m_1|g(n)| \leqslant m_1m_2|h(n)| = m|h(n)|$$
   so $f \in O(h)$. This shows domination is transitive.

   Since $6n^2 + 4n + 3$ and n^2 are both Big O of each other, domination is not antisymmetric, and since $n \in O(n^2)$ but $n^2 \notin O(n)$ it is not symmetric.

4. Finds the maximum entry in the array. Takes linear ie. $O(n)$ steps.

5. Sorts the array into ascending order. Takes quadratic time (compares each pair of numbers in the array, so takes $(n-1) + (n-2) + \ldots$ steps which is $O(n^2)$.

6. The best (prove it!) method is binary search: start at the middle of the array, and compare that entry with your target value. If less, move to the middle of the first half (and throw the right half away), and if greater, move to the middle of the right half. Repeat until you either locate the target value, or if the length of the remaining segment of the array has size 1. Count the number of times you make a "step". [6]

7. (a) Let S =pigeons, $m = |S| \geqslant 3$, and holes are labeled $0, 1$, send $x \in S$ to hole $[x]_2$. Since $\geqslant 3$ pigeons in 2 holes, by PHP some hole has two pigeons, call them x, y. If $[x]_2 = [y]_2 = 0$ then $x + y$ is even, and if $[x]_2 = [y]_2 = 1$ then $x + y \equiv 1 + 1 \equiv 0$ so again $x + y$ is even.

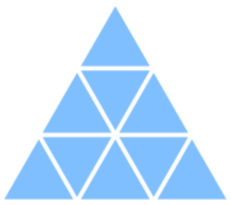
   (b) Let S =pigeons, $m = |S| \geqslant 7$, and holes are labeled $0, 1, 2$, send $x \in S$ to hole $[x]_3$, and put $k = 2$. Then since $nk = 2.3 = 6$ and $m > 6$ by GPHP some hole has more than $k = 2$ pigeons, so there are $x, y, z \in S$ with $[x]_3 = [y]_3 = [z]_3 = i$. Then $x + y + z \equiv 3i \equiv 0 \mod 3$ so $x + y + z$ is a multiple of 3.

8. $m = 51$ students=pigeons, $n = 5$ grades=holes, $k = 10$. Then by GPHP $m > kn = 50$ so some hole has more than 10 pigeons in it, so at least 11 students will get the same grade.

9. Pigeons=elements of S, pigeonholes:
$$\{0, p\}, \{1, p-1\}, \{2, p-2\}, \ldots \{\frac{p-1}{2}, 1 + \frac{p-1}{2}\}.$$
   Place each element of S in the subset containing it. There are $n = 1 + \frac{p-1}{2}$ holes and $m = 2 + \frac{p-1}{2}$ pigeons so some subset contains two elements. The sum of two elements in a subset is p.

10. Divide the triangle up into 9 equilateral triangles of side length $\frac{1}{3}$, these are the pigeonholes.

[6]Does a step include moving the pointer between cells, or just whenever you make a comparison of two integers?



Pigeons are the 10 points, which are placed in the mini-triangle in which they sit. If a point lies on a border between two, either always place the point into the triangle *above* (say), or you could make two points and place one in each. In the second case, you have $m \geqslant 10$ pigeons, and in the first case $m = 10$ to put into 9 holes, so by PHP some triangle contains two points. To finish, show that any two points in an equilateral triangle of side length $\frac{1}{3}$ lie within distance $\frac{1}{3}$ of each other.

11. Let x_i be the number of robots produced between day 1 and day i. Let $y_i = x_i + 15$. Then $1 \leqslant x_1 < x_2 < \cdots < x_{30} = 44$ and $16 \leqslant y_1 < y_2 < \cdots < y_{30} = 59$. Note that the sequence of x_i's is strictly increasing since each day at least one robot is produced, and same for the y_i's.

    Let the pigeonholes be the numbers from 1 to 59, and the pigeons the values for $x_1, \ldots, x_{30}, y_1, \ldots, y_{30}$, with the rule that x_i and y_i are placed in the hole corresponding to their value. So there are 60 pigeons and only 59 holes, so PHP says some hole has two values in it. But the x_i are all distinct, and the y_i are all distinct, so this hole must contain exactly one x_i and one y_j. So we have $x_i = y_j = x_j + 15$ and then we see $j < i$ and so on day j x_j robots have been produced, and then after $i - j$ days an additional 15 are produced.

    From
    `https://math.stackexchange.com/questions/2413861/how-to-prove-that-there-is-a-str`

12. Suppose the total alphabet size is N, then there are N^n strings of length exactly n, and we want an algorithm to send each of these strings to a distinct string of length $< n$. The number of shorter strings is

$$1 + N + N^2 + \cdots + N^{n-1}$$

which using induction you can show is equal to

$$\frac{N^n - 1}{N - 1}.$$

(You might have learned this as "geometric series".) Now here is the problem: this number is smaller than N^n, so by PHP if we have to send all N^n strings to something shorter, two strings will be sent to the same thing.

Proof that $\frac{N^n-1}{N-1} < N^n$: we assume $N \geqslant 2$ else we can't write much[7] so $\frac{1}{N-1} \leqslant 1$ so $N^n > N^n - 1 \geqslant \frac{1}{N-1}(N^n - 1)$.

[7]actually, we could write everything in unary. But we at least need a space symbol to separate words