A NOTE ON THE CIRCULAR JAIL EXERCISE

MANAS PATRA

In this short note, we discuss the lab exercise "circular jail". You were asked to write a program that enumerates the cell doors that are left open. A little reflection shows that a cell door $1 \leq n \leq 100$ is left open if and only if the number of factors of n is odd. In the first round, every door is opened (1 divides every number), in the second round every even-numbered door is closed and so on. For example, let $n = 12$. The status of the 12th door will change in rounds 1, 2, 3, 4, 6 and 12. The number of factors is 6 and you can easily see that it will be closed (obviously it will not be affected after round 12). For a positive integer n, let $F(n)$ denote the number of factors of n including 1 and n. Note the following facts.

(1) If two positive integers m and n are relatively prime (have no common factors) then $F(mn) = F(m)F(n)$. This is true because any factor of mn must be of the form ab where a is factor of m and b is factor of n . This is not true in general if m and n have common factors > 1.

(2) If $n = p^k$ where p is a prime number, then the factors are $1, p, p^2, \ldots, p^k$. Thus $F(n) = k + 1$. $F(n)$ is odd if and only if k is *even*, that is, n is a perfect square.

(3) Any positive integer can be (uniquely) written as a product prime powers. This means that

$$n = p_1^{k_1} p_2^{k_2} \cdots p_r^{k_r}$$

where $p_1, p_2, \ldots$ are distinct prime numbers and $k_i > 0$. So $p_i^{k_i}$ and $p_j^{k_j}$ are relatively prime. Hence

$$F(n) = F(p_1^{k_1}) F(p_2^{k_2}) \cdots F(p_r^{k_r})$$

Consequently, $F(n)$ is odd if and only if all the numbers $F(p_j^{k_j})$, $j = 1, 2, \ldots, r$ are odd. From the second item, this is possible if and only if *all* k_j are even. That is, n is perfect square. Hence a cell-door n will remain open if and only if n is perfect square.

A straightforward way of doing the exercise would involve two loops—the first one for each run of the drunken jailer and the second for implementing his crazy behaviour. So the time complexity is of order n^2 where n is the number of doors (100 in this case). But if we use the criterion that the door numbers must be perfect squares we have to only compute at most $\sqrt{n}$ times: find the squares of the numbers up to the largest whole number $\leq \sqrt{n}$. So doing a little maths, we have reduced complexity drastically!.

COMP9021 Principles of Programming

Date: Session 2, 2015.