CS30053 Foundations of Computing, Spring 2004

Practice exercises : Set 2

---

1. Which of the following estimates are true?

(a) $5n^2 \log n + 2(n^3/\log n)$ is $\mathrm{O}(n^3)$.

(b) $5n^2 \log n + 2n^3$ is $\Theta(n^3)$.

(c) $2^{\sqrt{n}}$ is $\Theta((\sqrt{2})^n)$.

(d) n^2 is $\Omega(n^3)$.

(e) 2^n is $\Omega(3^n)$.

(f) $(2n)!$ is $\Theta(n!)$.

(g) $\log((2n)!)$ is $\Theta(\log(n!))$.

(h) $1/n$ is $\Theta(1)$.

2. Let f and g be two real-valued functions of non-negative integer variables. We say that $f(n)$ is $\mathrm{o}(g(n))$, if $\lim_{n\to\infty} f(n)/g(n) = 0$, i.e., if for every $\epsilon > 0$ there exists $k \in \mathbb{N}$ such that $|f(n)| < \epsilon|g(n)|$ for all $n > k$. In this case, we also write $g(n) = \omega(f(n))$. Prove the following assertions:

(a) $2n + 3$ is $\mathrm{o}(5n^2 + 4n + 3)$.

(b) n is $\mathrm{o}(2^n)$.

(c) $2^{\sqrt{\lg n \lg\lg n}}$ is $\mathrm{o}(a^n)$ for any real $a > 1$.

(d) $2^{\sqrt{\lg n \lg\lg n}}$ is $\omega(n^d)$ for any $d \in \mathbb{N}$.

(e) $1/n$ is $\mathrm{o}(1)$.

(f) 2^{-n} is $\mathrm{o}(n^{-2})$.

(g) $f(n)$ is $\mathrm{o}(g(n))$ if and only if $f(n)$ is $\mathrm{O}(g(n))$ but $g(n)$ is not $\mathrm{O}(f(n))$.

(h) $f(n)$ is $\omega(g(n))$ if and only if $f(n)$ is $\Omega(g(n))$ but $g(n)$ is not $\Omega(f(n))$.

3. Why are the following *not* algorithms?

(a) **Input:** n integers $a_1, a_2, \ldots, a_n$.

Output: The smallest index i for which $a_i \geqslant a_{i+1}$.

Steps:

$i := 1$.
while $(a_i < a_{i+1})$ $i := i + 1$.
return i.

(b) **Input:** A positive integer n.

Output: The floating point sum $\frac{1}{1} + \frac{1}{2} + \cdots + \frac{1}{n-1}$.

Steps:

$s := 0$.
while $(n > 0)$ {
  $s := s + (1.0/(n-1))$.
  $n := n - 1$.
}
return s.

** **(c)** **Input:** Nothing.

Output: The smallest integer $n > 1$ for which $\frac{1}{1} + \frac{1}{2} + \cdots + \frac{1}{n}$ is an integer.

Steps:

$n := 1$, $a := 1$, $b := 1$.
repeat {
  $n := n + 1$.
  $a := an + b$.
  $b := bn$.
} until $b \mid a$.
return n.

(d) Write an algorithm for solving each of the above problems.

4. Determine what the following two algorithms compute. Also derive the worst-case time complexities of these algorithms. Assume that in each case the input is a sequence of n positive integers $a_1, a_2, \ldots, a_n$. For Part (b) assume that $a_1 \leqslant a_2 \leqslant \cdots \leqslant a_n$.

(a) for $i = 1$ to n set $b_i := 0$.
for $i = 1$ to n {
  for $j = 1$ to n {
    if $(a_j = a_i)$ then $b_j := b_j + 1$.
  }
}
$m := 0, j := 0$.
for $i = 1$ to n {
  if $(b_i > m)$ then { $m := b_i, j := i.$ }
}
return a_j.

(b) $m := 1, j := 1, c := 1, b := a_1$.
for $i = 2$ to n {
  if $(a_i \neq b)$ then { $c := 1, b := a_i.$ }
  else {
    $c := c + 1$.
    if $(c > m)$ then { $m := c, j := i.$ }
  }
}
return a_j.

(c) Describe how you can use the algorithm of Part (b) to solve the problem of Part (a) in $O(n \log n)$ time.

5. Suppose that the study of the properties of some objects located in the Andromeda Galaxy requires the computation of a formula $f(a_1, a_2, \ldots, a_n)$ defined for positive integers $a_1, a_2, \ldots, a_n$ as:

$$f(a_1, a_2, \ldots, a_n) := \sum_{i \in S} \left(\prod_{j=1}^{i} a_j^2 \right),$$

where S is the set $\{i \mid 1 \leqslant i < n \text{ and } a_i \leqslant a_{i+1}\}$. I supply the following algorithm for the computation of $f(a_1, a_2, \ldots, a_n)$.

$s := 0$.
for $i = 1$ to $n - 1$ {
  if $(a_i \leqslant a_{i+1})$ {
    $p := 1$.
    for $j = 1$ to i do $p := p \times a_j$.
    $s := s + p^2$.
  }
}
return s.

Compute the worst-case, average and best-case time complexity of my algorithm.

6. Suppose we want to compute for a given $n \in \mathbb{Z}^+$ the number of triples (i, j, k) with $1 \leqslant i \leqslant j \leqslant k \leqslant n$.

(a) Design an algorithm for this problem, that runs in $\Theta(n^3)$ time.

(b) Design an algorithm for this problem, that runs in $\Theta(n^2)$ time.

* **(c)** Design an algorithm for this problem, that runs in $\Theta(1)$ time. (**Hint:** Find a compact formula for the number of such triples.)

7. Consider the following modification of the binary search algorithm. Let $a_1, a_2, \ldots, a_n$ be a sorted sequence of integers. In order to determine if an integer a occurs in the list, divide the list in three parts of nearly equal sizes and determine in which part a may be located. Then divide this potential part into three subparts and so on. Since this search uses a three-way branching, one calls it t e r n a r y s e a r c h.

(a) Write the details of the ternary search algorithm.

(b) Compute the worst-case time complexity of the ternary search algorithm.

(c) If you are asked to choose between binary search and ternary search, which would you prefer. Why?

*** 8.** Prove that it is impossible to write a C program that takes as input a C program Q and an input I to Q and that determines whether Q on input I prints the string "Hello, world" as the first twelve printed characters.

9. Prove that the union of a countable number of countable sets is again countable. (**Hint:** Look at $\mathbb{N} \times \mathbb{N}$.)

*** 10.** Show that no set A can have bijective correspondence with its power set $\mathcal{P}(A)$. (**Hint:** Let $f : A \to \mathcal{P}(A)$ be a bijection. Look at the subset $S \subseteq A$ defined as $S := \{x \in A \mid x \notin f(x)\}$.)

11. Find the flaw in the following proof.

Theorem All horses are of the same color.

Proof Let there be n horses. We proceed by induction on n. If $n = 1$, there is nothing to prove. So assume that $n > 1$ and that the theorem holds for any group of $n - 1$ horses. From the given n horses discard one, say the first one. Then all the remaining $n - 1$ horses are of the same color by the induction hypothesis. Now put the first horse back and discard another, say the last one. Then the first $n - 1$ horses have the same color again by the induction hypothesis. So all the n horses must have the same color as the ones that were not discarded either time. •

12. The Fibonacci sequence is defined recursively as

$$\begin{aligned} F_0 &:= 0, \\ F_1 &:= 1, \\ F_n &:= F_{n-1} + F_{n-2} \text{ for } n \geqslant 2. \end{aligned}$$

Prove by induction that $F_n = \frac{1}{\sqrt{5}}\left[\left(\frac{1+\sqrt{5}}{2}\right)^n - \left(\frac{1-\sqrt{5}}{2}\right)^n\right]$ for all $n \in \mathbb{N}$.

13. Let F_n, $n \in \mathbb{N}$, denote the sequence of Fibonacci numbers defined in Exercise 12. By using induction on n prove the following assertions:

(a) $F_1^2 + F_2^2 + \cdots + F_n^2 = F_n F_{n+1}$ for all $n \in \mathbb{Z}^+$.

(b) If $A := \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}$, then $A^n = \begin{pmatrix} F_{n+1} & F_n \\ F_n & F_{n-1} \end{pmatrix}$ for all $n \in \mathbb{Z}^+$.

(c) $F_{m+n} = F_{m-1}F_n + F_m F_{n+1}$ for all $m \in \mathbb{N}$ and $n \in \mathbb{Z}^+$.

14. Let $H_n := \frac{1}{1} + \frac{1}{2} + \cdots + \frac{1}{n}$ for $n \geqslant 1$ (the h a r m o n i c n u m b e r s).

(a) Use calculus to show that $\ln(n+1) \leqslant H_n \leqslant \ln n + 1$ for all $n \geqslant 1$. (**Remark:** It is known that $(H_n - \ln n) \to \gamma$ as $n \to \infty$, where the constant $\gamma = 0.57721566\ldots$ is called E u l e r ' s c o n s t a n t.)

(b) Use induction to show that $H_1 + H_2 + \cdots + H_n = (n+1)H_n - n$ for all $n \geqslant 1$.

15. Let S be a subset of $\mathbb{N}$, such that 0 and 1 belong to S, and whenever n and $n + 1$ belong to S, so also does $n + 2$. Prove that $S = \mathbb{N}$.

16. Recursively define the following functions:

(a) $f_1 : \mathbb{N} \to \mathbb{N}$ given by $f_1(n) = 2n^3$ for all $n \in \mathbb{N}$.

(b) $f_2 : \mathbb{N} \to \mathbb{N}$ given by $f_2(n) = n^2 + 2n^3$ for all $n \in \mathbb{N}$.

(c) $f_3 : \mathbb{N} \to \mathbb{N}$ given by $f_3(n) = 3^{2n}$ for all $n \in \mathbb{N}$.

* **(d)** $f_4 : \mathbb{N} \to \mathbb{N}$ given by $f_4(n) = 2^n + 3^{2n}$ for all $n \in \mathbb{N}$.

(e) $f_5 : \mathbb{N} \to \mathbb{N}$ given by $f_5(n) = 3^{2^n}$ for all $n \in \mathbb{N}$.

(f) $\phi : \mathbb{Z}^+ \to \mathbb{Z}^+$ given by $\phi(n) := |\{a \mid 1 \leqslant a \leqslant n \text{ and } \gcd(a, n) = 1\}|$. (**Hint:** Use without proof the results that $\phi(p^e) = p^e - p^{e-1}$ for a prime p and an exponent $e \in \mathbb{Z}^+$ and that if m, n are coprime, then $\phi(mn) = \phi(m)\phi(n)$.)

(g) $\det : \mathcal{M} \to \mathbb{Z}$, where $\mathcal{M}$ is the set of all square matrices with integer entries, and where $\det(A)$ is the determinant of a matrix $A \in \mathcal{M}$.

(h) The binomial coefficients $\binom{}{} : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ defined by $\binom{n}{k} = \begin{cases} \frac{n!}{k!(n-k)!} & \text{if } k \leqslant n, \\ 0 & \text{otherwise.} \end{cases}$

* **(i)** $\mathcal{S} : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$, where the Stirling number $\mathcal{S}(n, k)$ stands for the total number of ways of partitioning a set of cardinality n into exactly k non-empty parts. (**Hint:** Let $A := \{a_1, a_2, \ldots, a_n, a_{n+1}\}$. A partition of A is of one of the two kinds: i) $\{a_{n+1}\}$ is itself a member of the partition, and ii) a_{n+1} lies in a bigger set in the partition.)

17. Recursively define the following sets:

(a) $S_1 := \{a \in \mathbb{N} \mid a \equiv 3 \text{ or } 5 \pmod 7\}$.

(b) $S_2 := \{(a, b) \in \mathbb{N} \times \mathbb{N} \mid a + b \text{ is odd}\}$.

(c) The set S_3 of all palindromes over the binary alphabet $\{0, 1\}$.

* **(d)** The set S_4 of all strings over the binary alphabet $\{0, 1\}$, that contain equal numbers of 0's and 1's.

** **(e)** The set S_5 of all strings over the binary alphabet $\{0, 1\}$ of the form $\alpha\alpha$ for some $\alpha \in \{0, 1\}^*$.

18. What is wrong in the following recursive definition of *sets*.

BASIS: The empty collection is a set.

INDUCTION: If S is a set and a is an object not present in S, then adding a to S gives a set.

(**Hint:** Infinity is not a natural number.)

19. Design recursive algorithms to compute the functions of Exercise 16. For Part (f) you may assume that you are given subroutines for primality testing and factorization of positive integers.

20. For notations refer to Exercise 17. Design recursive algorithms that compute the following:

(a) Given $k \in \mathbb{N}$, the cardinality of $\{a \in S_1 \mid a \leqslant k\}$.

(b) Given $k \in \mathbb{N}$, the elements of $\{a \in S_1 \mid a \leqslant k\}$.

(c) Given $(h, k) \in \mathbb{N}^2$, the cardinality of $\{(a, b) \in S_2 \mid (a, b) \leqslant (h, k) \text{ under the lexicographic ordering}\}$.

(d) Given $k \in \mathbb{N}$, the number of all palindromes over $\{0, 1\}$ of length $\leqslant k$.

(e) Given $k \in \mathbb{N}$, all palindromes over $\{0, 1\}$ of length $\leqslant k$, with each palindrome printed exactly once (but in any order of your convenience).

* **(f)** Given $k \in \mathbb{N}$, the number of all strings of S_4 of length $\leqslant k$.