Donald Knuth - Art of Computer Programming - Personal Solutions

Morgan Bruce

June 13, 2022

Contents

1 Basic Concepts **1**
1.1 Algorithms 1
1.2 Mathematical Preliminaries 3
1.2.1 Mathematical Induction 3

Chapter 1

Basic Concepts

1.1 Algorithms

1. [*10*] The text showed how to interchange the values of variables m and n, using the replacement notation, by setting $t \leftarrow m$, $m \leftarrow n$, $n \leftarrow t$. Show how the values of *four* variables (a, b, c, d) can be rearranged to (b, c, d, a) by a sequence of replacements. Try to use the minimum number of replacements.

Solution. This can be accomplished in 3 replacements: $a \leftrightarrow d$ which results in (d, b, c, a), $d \leftrightarrow b$ which results in (b, d, c, a), and finally $d \leftrightarrow c$ which results in (b, c, d, a).

We can prove this is the minimum number of replacements in this way: suppose a sequence of two replacements resulted in the exchange. Since every index of the rearrangement is not equal to the same index of the initial ordering, it must be the case that the pair of replacements act on different pairs of variables. But no pair of variables in the rearrangement can be swapped to recover the initial state without further replacements. Thus there must be more than two replacements, and since a method of using three replacements was demonstrated, it is the minimum number. □

2. [*15*] Prove that m is always greater than n at the beginning of step E1, except possibly the first time this step occurs.

Proof. Suppose $m < n$. Then the remainder of m/n must simply be $r \leftarrow m$, as the quotient is necessarily 0. Applying the next steps of the algorithm, we assign $m \leftarrow n$, $n \leftarrow r$. Thus after one step of the algorithm, $m \geq n$, and it suffices to check that the ordering remains after another step of the algorithm.

Suppose $m \geq n$. Then the remainder of m/n is some integer r such that $0 \leq r < n$. Applying the assignments, we have $m \leftarrow n$, $n \leftarrow r$, and thus since before assignments we had $r < n$, after assignments we have that $n < m$. So the ordering of m and n is preserved at the beginning of step E1 after the algorithm is applied. □

3. [*20*] Change Algorithm E (for the sake of efficiency) so that all trivial replacement operations such as "$m \leftarrow n$" are avoided. Write this new algorithm in the style of Algorithm E, and call it Algorithm F.

Solution. □

4. [*16*] What is the greatest common divisor of 2166 and 6099?

Solution. □

5. [*12*] Show that the "Procedure for Reading This Set of Books" that appears after the preface actually fails to be a genuine algorithm on at least three of our five counts! Also mention some differences in format between it and Algorithm E.

Solution. □

6. [*20*] What is T_5, the average number of times step E1 is performed when $n = 5$?

Solution. □

7. [*HM21*] Let U_m be the average number of times that step E1 is executed in Algorithm E, if m is known and n is allowed to range over all positive integers. Show that U_m is well defined. Is U_m in any way related to T_m?

Solution. □

8. [*M25*] Give an "effective" formal algorithm for computing the greatest common divisor of positive integers m and n, by specifying θ_j, ϕ_j, a_j, b_j as in Eqs. (3). Let the input be represented by the string $a^m b^n$, that is, m a's followed by n b's. Try to make your solution as simple as possible. [*Hint:* Use Algorithm E, but instead of division in step E1, set $r \leftarrow |m - n|$, $n \leftarrow \min(m, n)$.]

Solution. □

9. [*M30*] Suppose that $C_1 = (Q_1, I_1, \Omega_1, f_1)$ and $C_2 = (Q_2, I_2, \Omega_2, f_2)$ are computational methods. For example, C_1 might stand for Algorithm E as in Eqs (2), except that m and n are restricted in magnitude, and C_2 might stand for a computer program implementation of Algorithm E. (Thus Q_1 might be the set of all states of the matchine, i.e., all possible configurations of its memory and registers; f_2 might be the definition of single machine actions; and I_2 might be the set of initial states, each including the program that determines the greatest common divisor as well as the particular values of m and n.)

Formulate a set-theoretic definition for the concept "C_2 is a representation of C_1" or "C_2 simulates C_1." This is to mean intuitively that any computation sequence of C_1 is mimicked by C_2, except that C_2 might take more steps in which to do the computation and it might retain more information in its states. (We thereby obtain a rigorous interpretation of the statement, "Program X is an implementation of Algorithm Y.")

1.2 Mathematical Preliminaries

1.2.1 Mathematical Induction