5.4 Recursive Algorithms

5.4 pg 370 # 3

Trace Algorithm 3 when it finds gcd(8,13). That is, show all the steps used by Algorithm 3 to find gcd(8,13).

Algorithm 3

1. \( \text{gcd}(a, b : \text{nonnegative integers with } a < b) \)
2. \( \text{if } a = 0 \text{ then} \)
3. \( \text{return } b \)
4. \( \text{else} \)
5. \( \text{return } \text{gcd}(b \mod a, a) \)
6. \( \text{end if} \)
7. \{output is gcd(a, b)\}

5.4 pg 370 # 7

Give a recursive algorithm for computing \( nx \) whenever \( n \) is a positive integer and \( x \) is an integer, using just addition.

5.4 pg 370 # 9

Give a recursive algorithm for finding the sum of the first \( n \) odd positive integers.

5.4 pg 370 # 11

Give a recursive algorithm for finding the minimum of a finite set of integers, making use of the fact that the minimum of \( n \) integers is the smaller of the last integer in the list and the minimum of the first \( n - 1 \) integers in the list.

5.4 pg 371 # 45

Use a merge sort to sort \( b, d, a, f, g, h, z, p, o, k \) into alphabetic order. Show all the steps used by the algorithm.

Procedure 2 \( \text{mergesort}(L = a_1, \ldots a_n) \)

1. \( \text{if } n > 1 \text{ then} \)
2. \( m := \lceil n/2 \rceil \)
3. \( L_1 := a_1, a_2, \ldots, a_m \)
4. \( L_2 := a_{m+1}, a_{m+2}, \ldots, a_n \)
5. \( L := \text{merge(mergesort}(L_1), \text{mergesort}(L_2)) \)
6. \( \text{end if} \)
7. \{L is now sorted into elements in nondecreasing order\}