5.4 Recursive Algorithms

An algorithm is called recursive if it solves a problem by reducing it to an instance of the same problem with smaller input.

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 gcd(a, b : nonnegative integers with a < b)

1: if a = 0 then
2: return b
3: else
4: return gcd(b mod a, a)
5: end if

{output is gcd(a, b)}

gcd(8,13)
   gcd(13 mod 8 = 5, 8)
      gcd(8 mod 5 = 3, 5)
         gcd(5 mod 3 = 2, 3)
            gcd(3 mod 2 = 1, 2)
               gcd(2 mod 1 = 0, 1)
                  return 1

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.

Procedure 2 product(n : positive integer, x : integer)

1: if n = 1 then
2: return x
3: else
4: return x + product(n - 1, x)
5: end if

{output is nx}

5.4 pg 370 # 9

Give a recursive algorithm for finding the sum of the first n odd positive integers.
Procedure 3 \textit{oddSum}(n : positive integer)

1: \textbf{if} \ n = 1 \ \textbf{then}
2: \hspace{1em} \textbf{return} \ 1
3: \textbf{else}
4: \hspace{1em} \textbf{return} \ 2(n - 1) + 1 + \textit{oddSum}(n - 1)
5: \textbf{end if}

\{output is sum for 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.

Procedure 4 \textit{recursive_min}(n : positive integer, $a_1, a_2, a_3, \ldots, a_n$ : integers)

1: \textbf{if} \ n = 1 \ \textbf{then}
2: \hspace{1em} \textbf{return} \ a_1
3: \textbf{else}
4: \hspace{1em} \textbf{return} \ \textit{min}(a_n, \textit{recursive_min}(n - 1, a_1, a_2, a_3, \ldots, a_{n-1}))
5: \textbf{end if}

\{output is the minimum integer\}

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 5 \textit{mergesort}(L = a_1, \ldots a_n)

1: \textbf{if} \ n > 1 \ \textbf{then}
2: \hspace{1em} m := \lceil n/2 \rceil
3: \hspace{1em} L_1 := a_1, a_2, \ldots, a_m
4: \hspace{1em} L_2 := a_{m+1}, a_{m+2}, \ldots, a_n
5: \hspace{1em} L := \textit{merge}(\textit{mergesort}(L_1), \textit{mergesort}(L_2))
6: \textbf{end if}

\{L is now sorted into elements in nondecreasing order\}