Tutorial 13 solutions

1. We start by analyzing the amortized cost of the initial `minimum` operation. The real cost of this operation is at most n (in the case where every node of the tree has a left child but no right child – that is the tree is actually a list). The potential is initially 0 because there is no current node. After the call to `minimum`, the potential is $r_1 + (n - l_1) \leq 0 + (n - 0) = n$ (the `minimum` operation never follows an edge that goes to a right child). Thus the amortized cost of the `minimum` operation is at most $n + n = 2n$.

   We now analyze the amortized cost of the i^{th} `successor` operation. We break the analysis down into two subcases:

   - If the current node has a right child, then `successor` goes right once, and then left k times for some integer $k \geq 0$. The real cost of the operation is $k + 1$. The potential goes up by 1 for the move right, and then goes down by 1 for each of the left moves, and hence $\Phi(D_i) - \Phi(D_{i-1}) = 1 - k$. Therefore $cost_{am}(\text{operation } i) = (k + 1) + (1 - k) = 2$.
   - If the current node does not have a right child, then `successor` goes up from a right child k times, and then up from a left child once. The real cost of the operation is $k + 1$. The potential goes down by 1 for each of the first up moves from a right child, and then up by 1 for the move from a left child, and hence $\Phi(D_i) - \Phi(D_{i-1}) = 1 - k$. Therefore $cost_{am}(\text{operation } i) = (k+1) + (1-k) = 2$.

   The initial potential is 0, and so the worst-case running time of the sequence of $k + 1$ operations is at most $2n$ (the amortized cost of the `minimum` operation) $+ 2k$, which is in $O(n)$.

2. a. The result is clearly true for $i = 0$. Now assume that $\beta[i] \leq i$ for $i = 0, 1, \ldots, j-1$, and consider the j^{th} iteration of the `for` loop. Initially $b \leq j - 1$. Each iteration of the `while` loop decreases b (since $\beta[i - 1] \leq b - 1 < b$) and so by the end of the `while` loop, $b \leq j - 1$. Thus $\beta[i] \leq j - 1 + 1 = j$.

   b. Clearly $\Phi(D_0) = 0$. Consider now an iteration of the body of the `for` loop, where the `while` loop executes x times.

      The real cost of this iteration of the body of the `for` loop is in $\Theta(1) + x$, where the $\Theta(1)$ term covers all of the steps except for the `while` loop.

      Now, as argued in the answer to part (a), every iteration of the body of the `while` loop decreases the value of b by at least 1. Thus the potential goes down by at least x during the execution of the `while` loop. It then goes up by at most 1 after the `while` loop has finished. This means that

      $$\Phi(D_i) - \Phi(D_{i-1}) \leq -x + 1$$

      Therefore the amortized cost of the body of the `for` loop is

      $$\Theta(1) + x + (-x + 1)$$

which is in $\Theta(1)$. Since the `for` loop executes exactly `length[p]` $- 1$ times, we therefore conclude that the algorithm runs in $O($`length[p]`$)$ time.