ECE 606, Fall 2019, Assignment 2

Zhijie Wang, Student ID number: 20856733

zhijie.wang@uwaterloo.ca

September 16, 2019

1. Here are pseudo-code for required functions.

   - CreateList()
     ```
     L = CreateStack()
     return L
     ```

   - IsEmptyList(L)
     ```
     return IsEmptyStack(L)
     ```

   - Insert(L, i)
     ```
     Push(L, i)
     ```

   - Head(L)
     ```
     return Pop(L)
     ```

   - Delete(L, i)
     ```
     tmpStack = CreateStack()
     while (!IsEmptyStack(L))
           do tmp = Pop(L)
              if (tmp == i)
                 then continue
                 else Push(tmpStack, tmp)
     while (!IsEmptyStack(tmpStack))
           do Push(L, Pop(tmpStack))
     ```

   - Search(L, i)
     ```
     tmpStack = CreateStack()
     flag = false
     while (!IsEmptyStack(L))
           do tmp = Pop(L)
              Push(tmpStack, tmp)
              if (tmp == i)
                 then flag = true break
     while (!IsEmptyStack(tmpStack))
           do Push(L, Pop(tmpStack))
     return flag
     ```

2. *Proof.* Firstly, we give a claim that in the best solution that contains fewest coins, the number of 1-cent coin is smaller than 5, the number of 5-cents coin is smaller than 2, the number of 10-cents coin is smaller than 3 for $a \in \mathbb{N}$

Now we prove it. If we have 5 1-cent coins, we can replace them with 1 5-cents coin, which is fewer in number. And, if we have 2 5-cents coins, we can replace them with 1 10-cents coin, which is fewer. If we have 3 10-cents coins, we could replace them with 1 25-cents coin and 1 5-cents coin, which is fewer. So, in the best solution, the biggest amount we can reach with 10-cents, 5-cents and 1-cent coins are 24.

Then, we consider about the statement of Question 2. Suppose we have a solution A which use fewer coins than the greedy solution. Here we denote the number of 25-cents, 10-cents, 5-cents, 1-cent in greedy solution is n_1, n_2, n_3, n_4, and the number in solution A is m_1, m_2, m_3, m_4.

If $m_1 < n_1$, then we need to use 10-cents, 5-cents, 1-cent coins to replace every 1 25-cents coin, which is impossible in the best solution by the claim we have proved above. Then, if $m_2 < n_2$, we need to use 2 5-cents coins to replace every 1 10-cents coin, which is also impossible in the best solution. Similarly, if $m_3 < n_3$, we need to use 5 1-cent coins to replace every 1 5-cents coin, which is impossible, too. Finally, m_4 should equal to n_4. Therefore, it's impossible to find a solution A which use fewer coins than the greedy solution.

3. It's **True** that this new version of BinSearch terminates.

   Proof. We first observe that if BinSearch is invoked with $lo > hi$, then we return immediately in Line (6) as the while condition evaluates to false. For the case that BinSearch is invoked with $lo \leq hi$, if we return in Line (3) in that iteration, we are done, as the algorithm has terminated. Now suppose that when we enter an iteration of the while loop, we do so with lo, hi values of lo_1, hi_1, respectively. Suppose also that in that iteration, we do not return in Line (3). Thus, we are guaranteed to check the while condition again. This second time, suppose that the lo, hi values are $lo2$, $hi2$ respectively.

   We claim that $hi_2 - lo_2 < hi_1 - lo_1$.

   To prove this, let $m = \lceil \frac{lo+hi}{2} \rceil$. We observe that we need to consider two cases. (1) $lo_2 = m+1, hi_2 = hi_1$, and (2) $lo_2 = lo_1, hi_2 = m-1$.

   In Case (1):

$$\begin{aligned} hi_2 - lo_2 &= hi_1 - m - 1 \\ &= hi_1 - \left\lceil \frac{lo_1 + hi_1}{2} \right\rceil - 1 \\ &< hi_1 - \left(\frac{lo_1 + hi_1}{2} - 1 \right) - 1 \\ &= \frac{lo_1 + hi_1}{2} \\ &< hi_1 - lo_1 \end{aligned}$$

   In Case (2):

$$\begin{aligned} hi_2 - lo_2 &= m - 1 - lo_1 \\ &= \left\lceil \frac{lo_1 + hi_1}{2} \right\rceil - 1 - lo_1 \\ &< \left(\frac{lo_1 + hi_1}{2} + 1 \right) - 1 - lo_1 \\ &= \frac{hi_1 - lo_1}{2} \\ &< hi_1 - lo_1 \end{aligned}$$

   Thus, we have proven that $hi_2 - lo_2 < hi_1 - lo_1$. Thus, if we start out with $lo \leq hi$, then we either return in Line (3) in some iteration, or, if we do not, eventually $hi - lo < 0 \Rightarrow lo > hi$, and we exit the while loop and terminate.

4. a2p4.py