Homework 6

Wendi Chen

1 Q2: Coin changing

1.1 Problem a

In order to minimize the number of coins used, we can simply choose the coin with the largest denomination each time.

Algorithm 1 Pseudocode of the Greedy Algorithm to Make Change

Input: n: n cents

Output: a_i: the number of pennies, nickels, dimes and quarters respectively

```
set c_0=1 //penny
set c_1=5 //nickel
set c_2=10 //dime
set c_3=25 //quarter
set i=3
while i >= 0 do
    a[i] = n/c[i]
    n = n - a[i] × c[i]
    i = i - 1
end while
```

Proof 1.1 *We define the original problem as* $S(n)$. *For an optimal solution* $A = \{a_0, ..., a_3\}$, *we have* $n = a_0 \times c_0 + a_1 \times c_1 + a_2 \times c_2 + a_3 \times c_3$. *After that, we can assert that* $a_0 <= 4$,*otherwise we can replace 5 pennies with 1 nickel, which uses fewer coins. In the same way, we can prove that* $a_1 <= 1$, $a_2 <= 2$, *and these two equal signs will not be true at the same time. So,* $a_0 \times c_0 + a_1 \times c_1 + a_2 \times c_2 < 25$. *That ensures when we use exact divisor to obtain* a_3, *the globally optimal solution must be obtained. So the original problem is reduced to find the solution to* $S(n^{'})$ *using* a_2, a_1, a_0, *where* $n^{'} = n - a_3 \times c_3$. *We can use the same method to prove that greedy algorithm will generate the globally optimal solution* a_2, a_1, a_0. *Therefore, the algorithm always yields an optimal solution.*

1.2 Problem b

Proof 1.2 *Similar to problem a, we define the original problem as* $S(n)$. *For an optimal solution* $A = \{a_0, ..., a_k\}$, *we have* $n = a_0 \times c_0 + ... + a_k \times c_k, c_n = c^n$. *According to problem a, we have* $a_0, ..., a_{k-1} <= c - 1$. *So,* $a_0 \times c_0 + ... + a_{k-1} \times c_{k-1} <= (c-1) \times \frac{1-c^n}{1-c} = c^n - 1 < c^n$. *That ensures when we use exact divisor to obtain* a_k, *the globally optimal solution must be obtained. So* *the original problem is reduced to find the solution to $S(n^{'})$ using $c_{k-1}, ..., c_0$, where $n^{'} = n - a_k \times c_k$. We can use the same method to prove that greedy algorithm will generate the globally optimal solution $a_{k-1}, ..., a_0$. Therefore, the greedy algorithm always yields an optimal solution.*

1.3 Problem c

Let the set of coin denominations $D = \{10, 7, 1\}$. When we use greedy algorithm to make change for 14 cents. The answer will be one 10 and four 1s, which uses 5 coins. However, the globally optimal solution is two 7s, which uses only 2 coins.