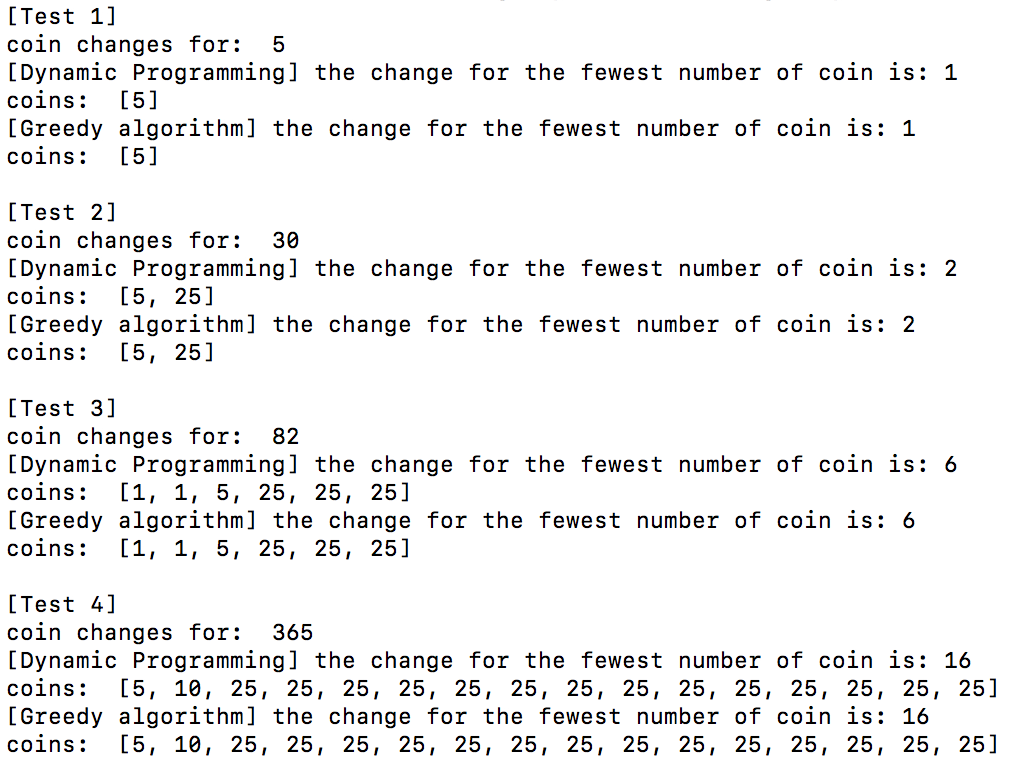
**Coin Changes Report**

**Kexin Ding**

**1. Source Code**

Please check coin\_changes.py

**2.Excution Results**



**3. Time Complexity Analysis**

For dynamic programming version, I use the method as the Bottom to Up method. Time complexity: T(n) = O(4n) + O(n) = O(4n) = O(n). 4 is the number of coin types: penny, nickel, dime, quarter.

For the greedy algorithm version, time complexity is T(n) = O(n). Because there is only one for loop.

**4. Answer of c**

a. We assume that we only use the first two denominations = 1 and = c. At most, we can use (c-1) \* because any value larger than c would be replaced by at least on c. This will reduce the total coin number by c-1. It means the remainder is greater than c and we can use only and . To satisfy the condition of fewer changes, we can use as many as we can before use .

b. We assume that we only use the first two denominations = 1 and . At most, we can use (c-1) \* because any value larger than would be replaced by at least on . This will reduce the total coin number by c-1. It means the remainder is greater than and we can use only and . To satisfy the condition of fewer changes, we can use as many as we can before use .

c. Consider a and b together, the greedy algorithm applies to all situations of this coin set. Hence greedy algorithm is optimal.