Marinel Tinnirello
12/6/2019
CMPT 435 - Assignment #11

1)

a) A = { 11_0, 7_1, 6_2, 48_3, 30_4, 12_5, 75_6 }

QuickSort(A, 0, 6)
p = Partition(A, 0, 6)
pivot = 48

A = { 11, 7, 6, 75, 30, 12, 48 } partition(A, 3, 6)
A = { 11, 7, 6, 12, 30, 75, 48 } partition(A, 3, 5)
A = { 11, 7, 6, 12, 30, 48, 75 } partition(A, 5, 6)

b)
```
partition ( A[], s, e )
        for i in [ 0, n-1 ]
                A[ i ] < pivot
                Insert ( A[ i ] -> A[] )
        insert ( pivot -> A[] )
        for i in [ 0, n - 1 ]
                A[ i ] >= pivot
                insert ( A[ i ] -> A[] )
```

c)
```
partition ( A[], s, e )
        p = 0, i = 1, j = n - 1
        while ( i < j )
                i++
        while ( A[ j ] >= pivot )
                j--
        if ( i < j )
                swap ( A[ i ], A[ j ] )
        swap ( A[ i ], A[ j ] )
```

2) Proof: (by contradiction) G doesn't minimize coins
Assumption: Assume G doesn't minimize coins
Compare: 1) a = a'; G uses same coins as 0
2) a > a'; G uses more coins than 0; CONTRADICTION
3) a < a'; G uses less coins than 0
O: a' - 1, b' - 2, c'
O: uses 2 coins less than 0; CONTRADICTION
b, b': 1) b = b'; c, d
2) b > b'; CONTRADICTION
1) O should not exist
2) O exists, O = G

3)

a) For a fractional output, I would take all of items 1 and 2, but take 2/3s of item 3. My steal would be $24, which is the most optimal.
b) For a 0-1 output, I would take all of items 2 and 3. My steal would be worth $22, which is the most optimal. A greedy algorithm does not best solve the knapsack problem.