Problem 1

1. T(i,j) is the length of the longest string Z(i,j) that appears to be both a substring of X that ends in X[i] and a subsequence in Y[1],..., Y[j]

2. Base Case: T(i, 0) = 0, T(0, j) = 0. 0 <= i <= n, 0 <= j <= m Recurrence: T(i, j) = T(i - 1, j - 1) + 1, if X(i) = Y(j) = T(i, j-1), if X(i) != Y(j) Where 0 < i <= n, 0 < j <= m

3. for i = 0 to n, T(i, 0) = 0 for j = 0 to m, T(0, j) = 0 for i = 1 to n for j = 1 to m if X(i) = Y(j), then T(i, j) = T(i - 1, j - 1) + 1, else T(i, j) = T(i, j -1) Return max(T(1, m), T(2, m) to T(n,m))

4. for i = 0 to n, T(i, 0) = 0 : O(n) for j = 0 to m, T(0, j) = 0 : O(m) for i = 1 to n for j = 1 to m : O(nm) Overall running time: O(nm)

Problem 2

1. T(i) = 1, if value i can make change by denominations x(1), ... , x(n), T(I) = 0, else

2. Base cases: T(0) = 1 Recurrence: In all x(1), ..., x(n), if exist x(j), that T(i - x(j)) = 1, and x(j) <= i, then T(i) = 1 else T(i) = 0 where 0 < i < n

3. T(0) = 1 for i = 1 to v: T(i) = 0 for i = 1 to v for j = 1 to n if x(j) < i, then T(i) = T(i - x(j)) if T(i) ==1: break return(T(v))

4. for i = 1 to v: T(i) = 0: O(v) for i = 1 to v for j = 1 to n : O(nv) Overall running time: O(nv)