Algorithm Assignment 3

Author: 410821305 Michael Hsueh

Q3.1.1: Explain why the problem is (or not) good for DP.

A3.1.1:

This problem is good for dynamic programming. The code from line 48 to line 55 (see A3-1 – Shopping list.cpp) shows how we do the dynamic programming. Here, we assume that each product can only be purchased once. By using the previous data in the array, we can calculate how much happiness we can get after including this product (loop i) by this budget (loop j). Since every j loop, we only need the previous j loop’s data in the array. Therefore, we can reduce the space complexity to O(n), while n represents the user budget. Every step running, we need the previous data in the array. Therefore, this is a good problem to solve by using dynamic programming.

Q3.1.2: Analyze the complexity of your algorithm.

A3.1.2:

1. Time complexity (dynamic programming part):

   Assume that there are k products in the shopping list, while the budget is n.

The time complexity is O(nk), either best case worst case.

2. Space complexity (dynamic programming part):

   Assume that the budget is n.

   The space complexity is O(n).

Q3.2.1: Explain why the problem is (or not) good for DP.

A3.2.1:

This problem is also good for dynamic programming. However, compare to the shopping list problem, the same area (array “rent_meters”) can be rented many times, the concept is common to the shopping list but the user can buy the same product many times. Therefore, the code is a little different. The key is at line 30 (see A3-2 – Isozaki Coast.cpp). By the code for(int j = rent_meter[i]; j <= n; j++), we can make the problem from only one product (concept in shopping list problem) to infinite products. Moreover, the rest are just the same as the shopping list problem. That is, compare when the length of the coast can afford the cost renting the area, then choose the maximum between after rented and before rented. We can also reduce the space complexity to O(n), while n represents the length of the coast that the user input. Since the current profit we want to know is based on the previous profit that we already knew. Therefore, this is a good problem to solve using dynamic programming.

Q3.2.2: Analyze the complexity of your algorithm.

A3.2.2:

1. Time complexity (dynamic programming part):

   Assume that there are k cost-area combinations and the length of the coast is n.

   The time complexity is O(nk).

2. Space complexity (dynamic programming part):

   Assume the length of the coast is n meters.

   The space complexity is O(n).