CPSC 3620—Fall 2023
Assignment 6

Due: December 6, 2023 11:55pm

Written Questions (50%)

Please hand in your assignment in Crowdmark.

All written questions are worth the same number of marks.

1. Recall the maximum sum subarray problem (see Assignments 2 and 3).

   (a) Suppose that the maximum subarray sum (S) for $A[0..k-1]$ as well as the maximum sum for a subarray that ends at $A[k-1]$ (E) are known. Show how to extend S and E to include one more element $A[k]$.

   (b) Using your answer above, show how to obtain an $\Theta(n)$ dynamic programming algorithm for the maximum sum subarray problem.

2. Recall the Floyd-Warshall algorithm. For this problem, we are interested in the number of paths between each pair of vertices i and j in a directed acyclic graph.

   (a) Suppose we know the number of paths between each pair of vertices where we restrict the intermediate vertices to be chosen from $1, 2, \ldots, k-1$, show how we can extend the result to allow vertex k as an intermediate vertex as well.

   (b) Give the algorithm. What is the complexity?

3. Consider the graph in Figure 1. Show how Prim's algorithm constructs a minimum spanning tree for this graph, with vertex 1 being the starting vertex. Show which edge is added at each step.

4. Consider the graph in Figure 1. Show how Kruskal's algorithm constructs a minimum spanning tree for this graph. Show which edge is added at each step.

5. Consider the graph in Figure 1. Show how Dijkstra's algorithm computes the shortest distances from vertex 1 to all other vertices. Show which vertex is added to the list of computed vertices as well as the updated distances to the remaining vertices at each step.

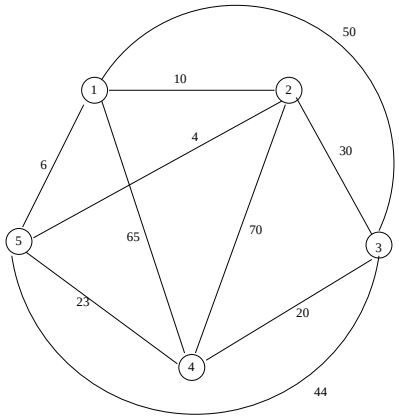

Figure 1: A graph.

Programming Question (50%)

For this problem, you will write two programs to compute the smallest number of coins needed to make up a certain sum. For example, if the denominations available are 1, 5, 10, 25, and 100, then we need 5 coins to make 146 (100 + 25 + 10 + 10 + 1).

Both programs should read in a list of up to 20 positive denominations as well as a target sum, and it will print the number of coins needed to make the target sum.

- You may assume that 1 is always one of the available denominations.
- You may assume that the denominations and the target sum does not exceed 100000.

1. Write your first program (`greedy.cc`) which uses a greedy strategy as follows:
   - Use a coin with the largest denomination that does not exceed the target sum, subtract it from the target sum.
   - Repeat until the target sum is 0.
2. Write your second program (`dp.cc`) which uses the dynamic programming strategy as follows:
   - The number of coins to make a sum of 0 is 0.
   - Assuming that the minimum number of coins needed to form all sums less than S is known, the minimum number of coins to form the sum S is:

     $$\min_{1 \le i \le n, D[i] \le S} (\text{min coins for } S - D[i]) + 1$$

     where $D[i]$ is the i-th denomination.

3. Try some denominations and target sums. Record at least 3 sets of inputs in which the two programs agree with each other. Record at least 3 sets of inputs in which the two programs do not agree with each other. Put this as part of your written assignment.

Write your programs in C++.

Submission

Submit your source file(s) and a Makefile. Your program should compile correctly simply by typing `make`. Name your executables `greedy` and `dp`.