About the solution

Let S denote the number of different speed limits occuring. It is clear that $S \leq 500$. Construct a new graph with N * S vertices, each one (n, v) corresponding to the state of being at crossing n and having speed v. There is a directed edge from vertex (n_1, v_1) to vertex (n_2, v_2) if and only if there is a road r from crossing n_1 to crossing n_2 and one of the following holds:

1. $V(r) = v_2$
2. $V(r) = 0$ and $v_1 = v_2$

If the weight of the edge is $L(r) / v_1$, then the original problem exactly corresponds to finding the shortest path in the new graph from (0, 70) to (D, v) with v arbitrary.

To find the shortest path, Dijkstra's algorithm can be used. The graph does not have to be constructed explicitly. In a typical instance, many of the vertices will not be reached since they have a much longer travel time than the destination. To solve all test cases, a priority queue must be implemented, for example by using a heap or a binary heap.

The test cases distinguishes between

1. a recursive solution
2. a clever recursive solution which saves the best path length for given crossing and speed
3. Dijkstra's shortest path with a normal array
4. Dijkstra's shortest path with a priority queue

The maximum number of crossings (150) and velocities (500) are chosen because even the fourth solution starts to take long time (~ 1 sec) in that region.