Manual Question:

Vertices : 7

Budget : 12

Edges: <source, destination, cost, time>

0,1,2,4

0,4,4,2

1,3,2,1

1,2,3,2

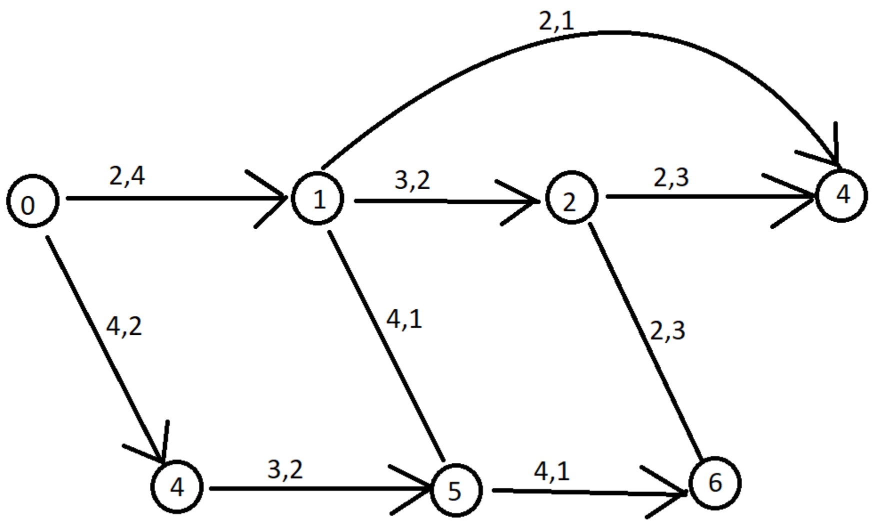
1,5,4,1

2,6,2,3

3,2,2,1

4,5,3,2

5,6,4,1



Queue: with list of expanded edges with the weight of the previous edges. The node also contains the indication if it was expanded or discarded

(0,0,0,0)(Used), (0,1,2,4)(Used), (0,4,4,2)(Used), (1,3,4,5)(Used), (1,2,5,6)(Used),(1,5,6,5)(Used), (3,2,6,6)(Garbage),(4,5,7,4)(Used), (2,6,7,9)(Used), (5,6,10,6)(Used), (5,6,11,5)(Used)

0 → (0,0,0,0)

1 → (0,1,2,4)

2→ (1,2,5,6)

3 → (1,3,4,5)

4 → (0,4,4,2)

5 → (1,5,6,5) , (4,5,7,4)

6 → (2,6,7,9), (5,6,10,6), (5,6,11,5)

Program Explanation:

First the file is read, leaving the first line as it is blank, and the values of the vartices and the edges are populated.

Adding the initial weight on the source as (source, source, 0, 0). All reachable vertices are relaxed to find the shortest path. The elements with higher cost than the budget and higher cost or time than an existing path are discarded. Then the element with least cost and time is picked from the queue to relax. The relaxed paths are added to the path array of vertices. The shortest or the cheapest path can then be found by going over the list of paths in the list corresponding to the destination vertex in the path array.

Finally, all the costs and time pairs are returned for reaching the destination from the source with the path taking the least time in the given budget.

Time complexity:

For populating the Paths, log E for inserting in the priority queue and E iterations based on the relaxation of each vertex. Therefore, the time is O(E log E)

Fetching the path can go up to E in the worst case.

Hence the total time complexity is O(E log E)