Explanation of Input:

N M

7 9

1 2 4

..

M

N is the number of nodes, M is the number of edges. Each line with the slope will describe 1 edge. The edges described as a b w meaning edge from a to b with weight w. So, in this case we have 7 nodes and 9 edges. {1 2 4} this line says that there is an edge from 1 to 2 with weight 4.

Code:

We have an array of 100 places. It is the number of nodes we will be having. If we will be having 1000 we will write there 1000. Then we have *find* function that finds the highest ancestor of x. So, the highest ancestor of x it is going to be the father of itself if it isn’t it will return find father x. *Unite* is going to unite 2 trees. We have 2 nodes, one from each tree: x and y. We find the fathers of each node and make the father of the father of x to be the father of y. So, in this way we connect 2 trees. In *main* we initialize the father’s array which we will use in disjoint sets and then we declare the variables to load the input. We have integers *number\_of\_nodes*, *number\_of\_edges* and *a, b, weight* as explained before. Then we have *edges* vector that it is going to pair. In the first place of the first pair will have the weight and then we will have another pair which will hold the first node and the second node. Then we start by loading the input we load *number\_of\_nodes* and *number\_of\_edges* then we read them in and then we make a for loop from 0 to *number\_of\_edges* – 1 to read the edges. Then we push in our vector make pair first part is going to be the weight. Second part is going to be another pair.

Now we begin with Kruskal’s algorithm. Firstly, we declare the variables for MST. We have weight of the MST, the edges it has and the next\_index that we will check in vector edges. After that we sort the list of edges. Then we have while loop. We need the MST edges to be less to be smaller than number\_of\_nodes – 1. We will need the index to be lower than number\_of\_edges so we will never get out of our edges vector. Then we break the edge into the 3 integers they describe it. So, we have in the second pair the first number is going to be a. In the second pair the second number is going to be b and the first number we see in each place is going to be the weight of our edge. Now we check if the edge is ok to be included in the MST that is where it doesn’t create cycle and that means a and b are in different trees. We check if the ancestor of A it’s not the same as the ancestor of b that means they are not in the same tree. Then we add the weight of the edge to the MST weight variable. Then we print it, and we count one more edge inside our MST. After if statement ends, we increase the index of the edge we will be checking. And we repeat the loop again until we have n -1 edges in our tree then we present the weight and that is the end of program.