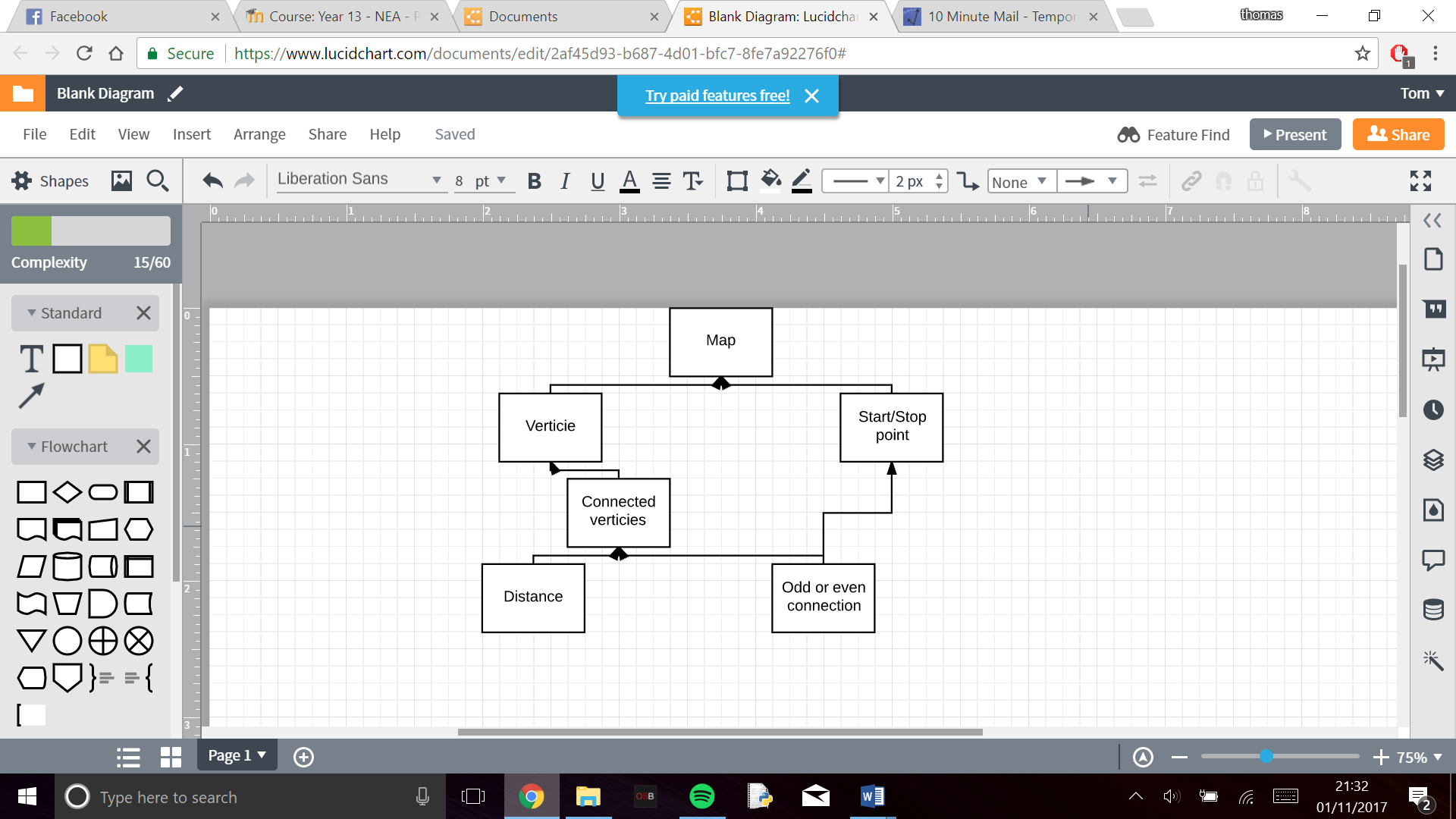
**Hierarchical chart:**



**Pseudo code:**

True

While True:

Vertices = 0

Input (“Are there more vertices?”)

If Yes:

Vertices = Vertices +1

If No:

False

For each vertices:  
 Input AmountOfConnections

For AnountOfConnections:  
 Input WhichConnectedTo

Input Distnace from connection

For each vertices:

If AmountOfConnection%2 = 1:

AddToRepeatVerticie

Input start vertice

Input finish vertice

If start and finish vertice in AddToRepeatVertice:

Apply Route inspection algorithm

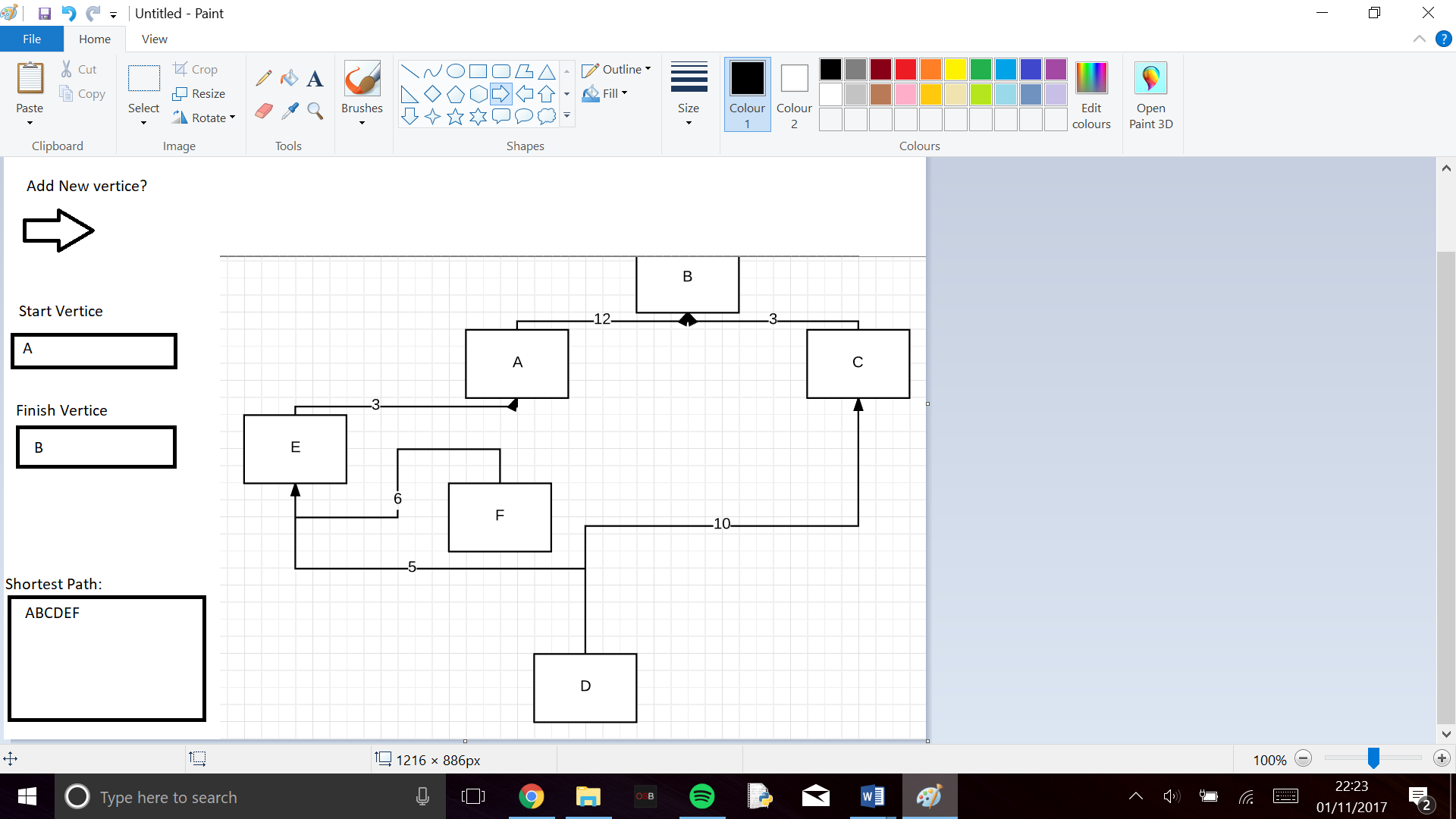
Else:

Apply Dijkstra’s to start and or finish

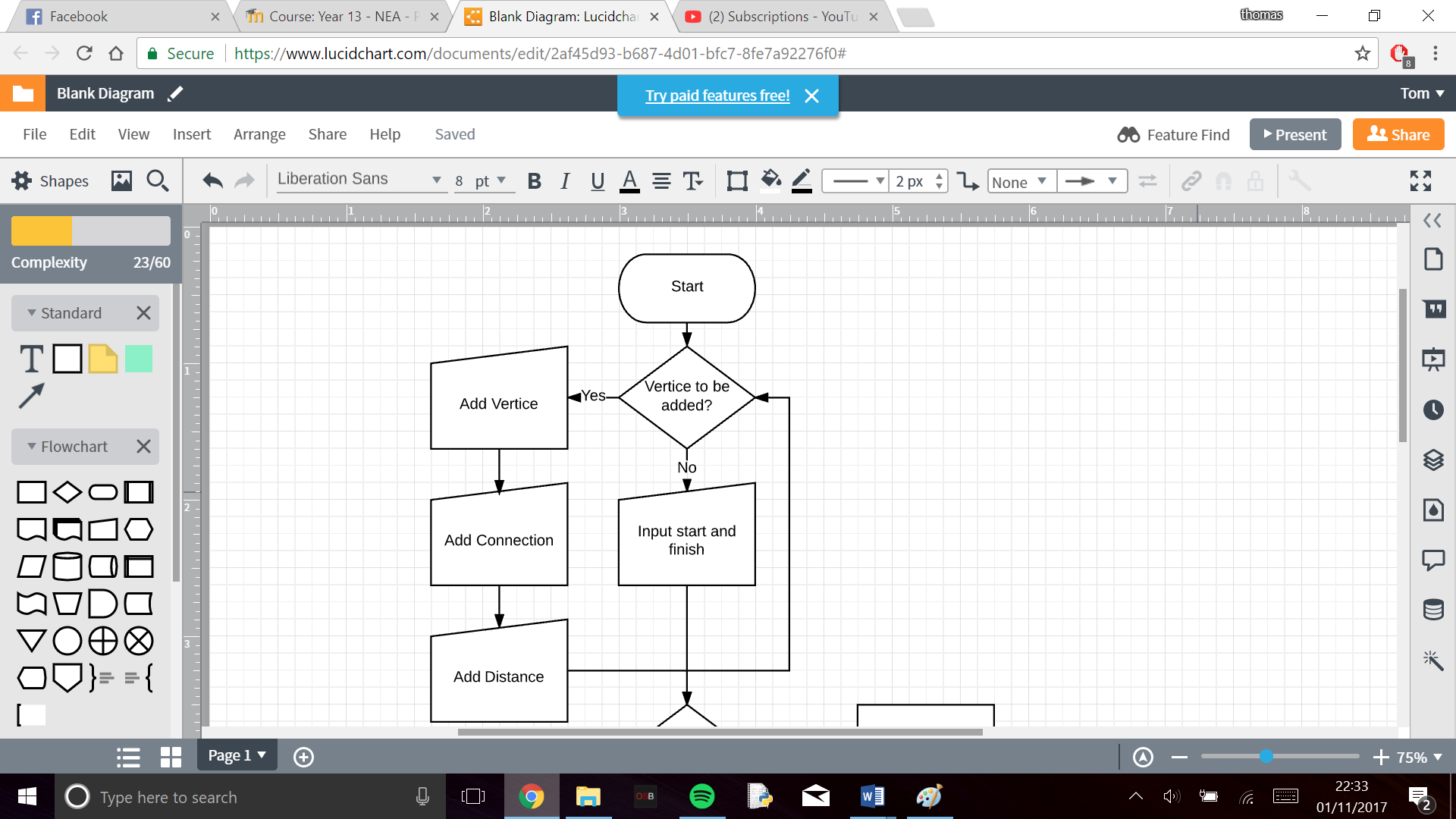
Apply Route inspection algorithm

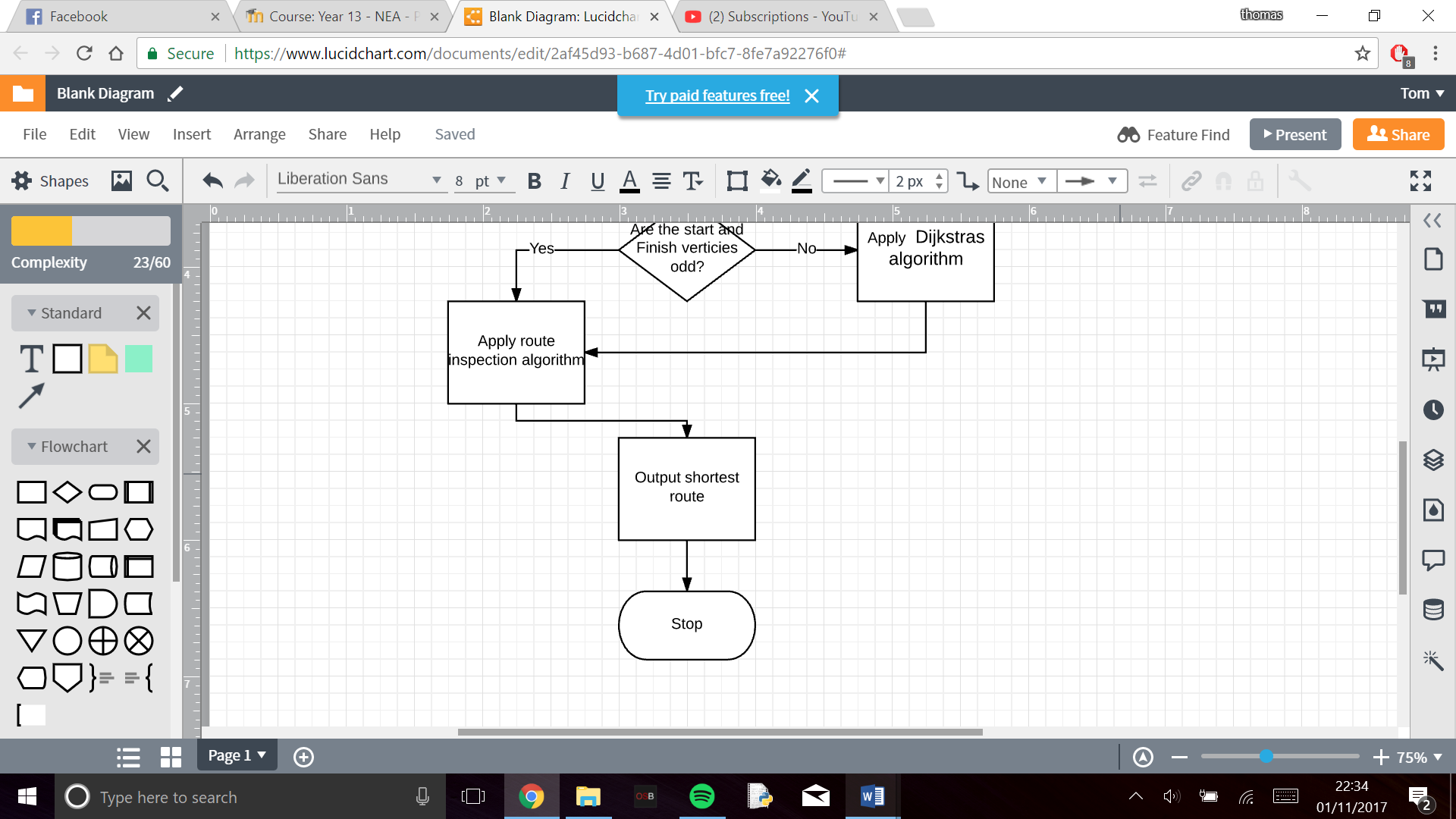
Print ShortestPath

**User interface:**



**Flowchart:**





**Algorithms:**

Route inspection:

* Identify any vertices with odd valency
* Consider all possible complete pairings of these vertices
* Select the complete pairing that has the least sum
* Add a repeat for the arcs indicated by this pairing to the network

Dijkstras:

* Use start vertex (S) and final vertex (0)
* Record a working variable at every vertex, Y that is directly connected to the vertex, X, that has just recived its final label.
  + Working value of Y = final value at X + weight of acr XY
  + If there is already a working value at Y, it is only replaced if the new value is smaller
  + Once a vertex has a final lable it is not revisited and its working values are no longer considered
* Look at the working values at all vertices without final lables. Select the smallest working value. This now becomes the final label at the vertex. (If two verticies have the same smallest working value either may be given as its final lable first)
* Repeat steps 2 and 3 until the destination vertex, T, recives its final label
* To find the shortest path, trace back from T to S. given that B already lies on the route, include arc AB whenever final label of B – final lable of A = weight of arc AB