**3.6 Homework Task Submission**

|  |  |
| --- | --- |
|  |  |
| **Answer all parts of this Homework task and submit your work into VSV Online as PDF and SnapApps/Edgy .xml or Python3/Trinket code files. This homework is required to demonstrate learning outcomes to a satisfactory standard.**  **For this Homework Submission 2 Files are expected:**  **• a PDF file with text responses to Problem, parts a, b, c, e**  **• an exported coding file in .xml from SnapApps/Edgy or using Python3/Trinket addressing Problem, part d** |
|  |

|  |  |  |
| --- | --- | --- |
| This weighted graph G=(V,E) V(G)={0,1,2,3,4,5,6,7} is available as  **3.6 SAT Folio CreateGraph.xml** ready to import into SNAP/Edgy <https://snapapps.github.io/edgy/app/edgy.html>.    **Figure 1:** |  | |
|  | **Edges E(G)** | **Weight** |
| **Prims Algorithm in English**  **Start with any node as a single-node tree T,**  Attach a new edge to a single growing tree T=(V,E) at each step until there are |V| nodes in T:   * always taking the minimum-weight edge that connects a node on the tree T to a node not yet on the tree.   **Prims Algorithm in Pseudocode**  Input: ***G=(V,E*)** a weighted graph  //Start the MST by selecting any vertex in the Graph  Initialize MST: **T=**(***V*new= {}, *E*new= {} )**  Repeat until (number of nodes in the MST T = **|V| )**  Choose **minimal weighted edge** u-v where ***u*** is  in ***Vnew***and ***v*** is not  Add node ***v*** to ***V*new**  Add edge ***u-v*** to ***E*new**  End Repeat  Output: **T=(*V*new**, ***E*new)** | 4 - 5 | 3.5 |
| 4 - 7 | 3.7 |
| 5 - 7 | 2.8 |
| 0 - 7 | 1.6 |
| 1 - 5 | 3.2 |
| 0 - 4 | 3.8 |
| 2 - 3 | 1.7 |
| 1 - 7 | 1.9 |
| 0 - 2 | 2.6 |
| 1 - 2 | 3.6 |
| 1 - 3 | 2.9 |
| 2 - 7 | 3.4 |
| 6 - 2 | 4.0 |
| 3 - 6 | 5.2 |
| 6 - 0 | 5.8 |
| 6 - 4 | 9.3 |

**Tasks 1 to 4 inclusive are required to complete this Folio**

1. Run Prims Algorithm by hand on the graph shown in Figure 1 above, and complete the table above showing step by step creation of the Minimum Spanning Tree (MST) by filling in the columns of the table below.

|  |  |  |  |
| --- | --- | --- | --- |
| **Nodes in T (MST)** | **Edges with exactly one endpoint in T**  **(sorted by weight lightest first)** | **New Edge Selected by Prims** | **Highlight the growing MST** |
| **Select a starting node: 1** | **[(1, 7), 1.9], [(1, 3), 2.9], [(1, 5), 3.2], [(1, 2), 3.6]** | (1,7) |  |
| **1, 7** | **[(7, 0), 1.6], [(7, 5), 2.8], [(1, 3), 2.9], [(1, 5), 3.2], [(7, 2), 3.4], [(1, 2), 3.6], [(7, 4), 3.7]** | (7,0) |  |
| **1, 7, 0** | **[(0, 2), 2.6], [(7, 5), 2.8], [(1, 3), 2.9], [(1, 5), 3.2], [(7, 2), 3.4], [(1, 2), 3.6], [(7, 4), 3.7], [(0, 4), 3.8], [(0, 6), 5.8]** | (0,2) |  |
| **1, 7, 0, 2** | **[(2, 3), 1.7], [(7, 5), 2.8], [(1, 3), 2.9], [(1, 5), 3.2], [(7, 4), 3.7], [(0, 4), 3.8], [(2, 6), 4.0], [(0, 6), 5.8]** | (2,3) |  |
| **1, 7, 0, 2, 3** | **[(7, 5), 2.8], [(1, 5), 3.2], [(7, 4), 3.7], [(0, 4), 3.8], [(2, 6), 4.0], [(3, 6), 5.2], [(0, 6), 5.8]** | (7,5) |  |
| **1, 7, 0, 2, 3, 5** | **[(5, 4), 3.5], [(7, 4), 3.7], [(0, 4), 3.8], [(2, 6), 4.0], [(3, 6), 5.2], [(0, 6), 5.8]** | (5,4) |  |
| **1, 7, 0, 2, 3, 5, 4** | **[(2, 6), 4.0], [(3, 6), 5.2], [(0, 6), 5.8], [(4, 6), 9.3]** | (2,6) |  |
| **1, 7, 0, 2, 3, 5, 4, 6** | **[]** | Done! |  |
| Total Cost of the Minimum Spanning Tree is: 18.1 | | | |

**To complete this SAT Folio task you are required to import an xml file into Edgy for this SAT task from VSV online, 3.6 SAT Folio CreateGraph.xml is ready to import into SNAP/Edgy** [**https://snapapps.github.io/edgy/app/edgy.html**](https://snapapps.github.io/edgy/app/edgy.html)**.**

1. Implement Prim’s Algorithm in Edgy. Show evidence of testing of your Edgy implementation on the graph shown in Figure 1.
2. Export your project to an \*.xml file with your initials as part of the name eg. **GG**Prims.xml
3. Give an argument of justification in your own words why Prim’s works to find a MST. Describe if there are any cases where it does not work or any cases where the solution is not unique.

|  |
| --- |
| **Prim’s works because it always starts with an MST, and each next step is always part of the MST.** |
| **The first node has to be part of the MST, because all nodes are in the tree** |
| **If the graph is partitioned into two sets, those already in the tree and those not, the edge added must be the shortest one between these two divided sets, because any other will increase the cost.** |
| **This is exactly what prims chooses, and ∴ the tree generated will be a subgraph of the MST** |
| **This holds until all the nodes are included in the MST, Prim’s will always generate an MST.** |

**For this Submission 2 Files are expected:**

* **1 PDF file with text responses to Task 1 & 4**
* **1 .xml Edgy export file addressing Task 2**