

NAME:P.HARI HASSAN

REG.NO:192210633

DATE:15/05/23

1.C PROGRAMM FOR PRIM'S ALOGRITHM

The screenshot displays a C++ IDE with the file `prim's algorithm.cpp` open. The code implements Prim's Algorithm to find the Minimum Spanning Tree (MST) of a weighted undirected graph. It uses a priority queue (min-heap) to select the next edge to add, ensuring no cycles are formed and no vertex has a degree greater than 2. The output window shows the following input and results:

```
Enter the number of vertices: 4
Enter the source vertex: 1
Enter the number of edges: 5
Enter the vertices pair along with cost: 1 2 1
Enter the vertices pair along with cost: 2 3 1
Enter the vertices pair along with cost: 3 4 1
Enter the vertices pair along with cost: 4 1 4
Enter the vertices pair along with cost: 1 3 3

***FINAL TABLE***

Vertex Known Dv Pv
-----
1 1 0 0
2 1 1 1
3 1 1 2
4 1 1 3

Minimum cost of spanning tree is 3

Process exited after 69.04 seconds with return value 0
Press any key to continue . . .
```

The code in the background includes standard C++ headers, defines a constant for infinity, and uses a linked list to represent the graph's edges. It includes functions for building the graph, calculating the MST, and finding the minimum cost.


```
C:\Users\HP\Documents\prim's algorithm.cpp - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
(globals)
Project Classes Debug prim's algorithm.cpp
181 list temp;
182
183 print("*****FINAL TABLE*****");
184 print("-----");
185 print("Vertex\Known\Dist\Path");
186 print("-----");
187
188 for (i = 1; i <= N; i++)
189 {
190     print("Vertex\Known\Dist\Path", i, T[i].known, T[i].dist, T[i].path);
191 }
192
193 void print(int N)
194 {
195     int v, w, i, c = 0, min = infinity;
196
197     while (c <= N)
198     {
199         for (i = 1; i <= N; i++)
200         {
201             if (T[i].known == 0)
202             {
203                 if (min > T[i].dist)
204                 {
205                     min = T[i].dist;
206                     v = i;
207                 }
208             }
209         }
210
211         T[v].known = 1;
212         Totalcost += T[v].dist;
213         cout << endl;
214         for (i = 1; i <= N; i++)
215         {
216             w = adjacent[i];
217             if (T[w].known == 0 && cost < T[w].dist)
218             {
219                 T[w].dist = cost;
220                 T[w].path = v;
221             }
222             min = infinity;
223         }
224         c++;
225     }
226
227     int main()
228     {
229         int N, s;
230         print("Enter the number of vertices: ");
231         scanf("%d", &N);
232         print("Enter the source vertex: ");
233         scanf("%d", &s);
234         initTable(s, N);
235         printTable(N);
236         print("Minimum cost of spanning tree is %d", Totalcost);
237         return 0;
238     }
239 }
```

Compiler Resources Compile Log Debug Find Results

Line: 165 Col: 2 Sel: 0 Lines: 165 Length: 2971 Insert Done parsing in 0.469 seconds

13:02 15-05-2023