a)

Pseudocode for Prim’s Algorithm

1. **Input**:
    - A graph G(V,E) is represented as an adjacency matrix or adjacency list, where V is the set of vertices and E is the set of edges.
    - A starting vertex r.
2. **Output**:
    - The MST represented as a set of edges.

Algorithm:

```
PRIM(G, r):
1. Initialize MST as an empty set.
2. Initialize a min-priority queue Q with all vertices and their key values (infinity for all except r, which is 0).
3. While Q is not empty:
   a. Extract the vertex u with the smallest key from Q.
   b. Add u to the MST.
   c. For each vertex v adjacent to u:
      i. If v is in Q and the weight of edge (u, v) is less than the current key of v:
         - Update v's key to the weight of edge (u, v).
         - Update v's parent to u.
4. Return the MST.
```

---

b)

Time Complexity

1. **Using an adjacency matrix:**
    - Extracting the minimum vertex takes O(V^2).
    - Time complexity: O(V^2).
2. **Using an adjacency list with a min-heap:**
    - Extracting the minimum vertex takes O(logV).

- Updating adjacent vertices takes O(logV) per edge.
- Time complexity: O(ElogV).