```
def BFS(g, s, discovered):
  """Perform BFS of the undiscovered portion of Graph g starting at Vertex s.

  discovered is a dictionary mapping each vertex to the edge that was used to
  discover it during the BFS (s should be mapped to None prior to the call).
  Newly discovered vertices will be added to the dictionary as a result.
  """
  level = [s]                        # first level includes only s
  while len(level) > 0:
    next_level = []                  # prepare to gather newly found vertices
    for u in level:
      for e in g.incident_edges(u):  # for every outgoing edge from u
        v = e.opposite(u)
        if v not in discovered:      # v is an unvisited vertex
          discovered[v] = e          # e is the tree edge that discovered v
          next_level.append(v)       # v will be further considered in next pass
    level = next_level               # relabel 'next' level to become current

def BFS_complete(g):
  """Perform BFS for entire graph and return forest as a dictionary.

  Result maps each vertex v to the edge that was used to discover it.
  (vertices that are roots of a BFS tree are mapped to None).
  """
  forest = {}
  for u in g.vertices():
    if u not in forest:
      forest[u] = None            # u will be a root of a tree
      BFS(g, u, forest)
  return forest
```