Name: Shubham S. Yadav
Date: 14-01-21

Algorithm for Problem 1: Vehicle Parking

```
function VehicleParking {

    print "enter the number of nodes";
    take input from user and store in nodes variable;

    print "enter the number of edges";
    take input from user and store in edges variable;

    print "enter the parking fees";
    take input from user and store in fees variable;

    FOR(i := 1 to number of nodes)  DO  {

        print "enter the capacity for {i} location";
        take input from user and store in capacity array/vector;
    }
    ENDFOR

    FOR(i := 0 to number of edges) DO {

        print "enter v{i}";
        take input from user;

        print "enter u{i}";
        take input from user;

        print "enter w{i} (cost of the path)";
        take input from user;

        # create a graph matrix from those edges and their costs

        graphMatrix[v][u] := w;
        graphMatrix[u][v] := w;
    }
    ENDFOR

    print "enter the number of vehicles"
    take input from user and store in totalVehicles variable

    create a priority queue/heap named pqueue with (v, u), w

    i := 1;

    WHILE(i <= totalVehicles and not pqueue.empty()) {

        # visiting first node
        v := pqueue.top().second;
        u := pqueue.top().first;

        # removing
```

```
        pqueue.pop();

        IF(not visited[v]) THEN {

            # visit the non-visited
            visited[v] := true;

            WHILE (i <= totalVehicles and capacity[v]--) {

                # adding total cost to the answer
                answer[i] := u + fees;
                i := i + 1;
            }
            ENDWHILE

            FOR (j := 0 to graphMatrix.size()) DO {

                IF (not visited[graphMatrix[v][j].second]) THEN {

                    pqueue.push(
                        graphMatrix[v][j].first + u,
                        graphMatrix[v][j].second
                    );
                }
                ENDIF
            }
            ENDFOR
        }
        ENDIF
    }
    ENDWHILE

    FOR (i := 1 to totalVehicles) DO {

        print "Cost (Path + Parking fees) for vehicle number {i} is: {answer[i]}";
    }
    ENDFOR
}
```

Algorithm for Problem 2: 8 Queens Problem

```
function CheckPlace(k, place) {

    FOR(i := 1 to(k - 1)) DO {

        IF((arr[i] = place) OR (abs(arr[i] - place) = abs(i - k))) THEN {

            return false;
        }
        ENDIF
    }
    ENDFOR
}

function EightQueens(k, n) {

    FOR(place := 1 to n) DO {

        IF(CheckPlace(k, place)) THEN {

            arr[k] := place;

            IF(k = n) THEN {

                print (arr[1:n]);
            }
            ENDIF
        }
        ELSE {

            EightQueens(k + 1, n)
        }
        ENDIF
    }
    ENDFOR
}
```