1. HashMap (Custom Implementation)

cpp
CopyEdit

```
HashMap<QString, StadiumInfo> stadiumMap;
HashMap<QString, QVector<Souvenir>> souvenirMap;
```

Operations and Big O:

- **Insert:** O(1) average case
- **Search:** O(1) average case
- **Delete:** O(1) average case
- **Worst Case:** O(n) (if there are many collisions)

Usage:
Quick lookups of stadium information and souvenirs.

Example:

cpp
CopyEdit

```
stadiumMap.get(teamName, info);
```

---

2. QVector (Qt's Dynamic Array)

cpp
CopyEdit

```
QVector<QString> order;
QVector<QString> stops;
QVector<Souvenir> souvenirs;
```

Operations and Big O:

- **Access:** O(1)

- **Insert at End:** O(1) amortized
- **Insert at Beginning/Middle:** O(n)
- **Delete:** O(n)
- **Search:** O(n)

Usage:
Storing ordered sequences of stadiums in paths and lists of souvenirs.

Example:

cpp
CopyEdit

```
order.append(stadium);
```

---

3. QMap (Qt's Balanced Binary Tree)

cpp
CopyEdit

```
QMap<QString, double> distances;
QMap<QString, QString> previous;
```

Operations and Big O:

- **Insert:** O(log n)
- **Search:** O(log n)
- **Delete:** O(log n)
- **Range Queries:** O(log n + k), where k = number of elements in range

Usage:
Used in Dijkstra's algorithm for maintaining sorted distances and previous nodes.

Example:

cpp

CopyEdit

```
distances[stadium] = std::numeric_limits<double>::infinity();
```

---

4. QSet (Qt's Hash Set)

cpp
CopyEdit

```
QSet<QString> unvisited;
QSet<QString> visited;
```

Operations and Big O:

- **Insert:** O(1) average case
- **Search:** O(1) average case
- **Delete:** O(1) average case
- **Worst Case:** O(n) (if there are many collisions)

Usage:
Tracking visited and unvisited nodes in graph algorithms.

Example:

cpp
CopyEdit

```
unvisited.insert(stadium);
```

---

5. Adjacency Matrix (Custom Implementation)

cpp
CopyEdit

```
QMap<QString, QMap<QString, double>> adjMatrix;
```

Operations and Big O:

- **Edge Lookup:** O(1)
- **Edge Insertion:** O(1)
- **Edge Deletion:** O(1)
- **Space Complexity:** O(V^2), where V = number of vertices

Usage:
Representing the graph of stadiums and their distances.

Example:

cpp
CopyEdit

```
adjMatrix[from][to] = distance;
```

Summary of Data Structure Choices

- **HashMap:** Fast lookups for stadium and souvenir data
- **QVector:** Efficient for ordered sequences and paths
- **QMap:** Sorted access for Dijkstra’s algorithm
- **QSet:** Fast membership testing for graph algorithms
- **Adjacency Matrix:** O(1) edge lookups for the graph