

Unordered Associative Containers Solutions

- Explain (briefly!) why unordered associative containers are more efficient than the ordered equivalents
 - An ordered container must keep its elements in sorted order
 - An unordered container avoids the overhead of sorting elements
- Which data structure is typically used to store unordered containers?
 - Hash table

- What output would you expect to see from the code shown on the next page?
 - The output is displayed in an order which depends on the hashing algorithm used by the container. On my system, the output was
 - Hareesh has a score of 77
 - Graham has a score of 72
 - Graham has a score of 66
 - Graham has a score of 78
 - Grace has a score of 66
- Convert this code to a full, working program

```
#include <unordered_map>
```

```
unordered_multimap<string, int> scores;
```

```
scores.insert( {"Graham", 78} );
```

```
scores.insert( {"Grace", 66} );
```

```
scores.insert( {"Graham", 66} );
```

```
scores.insert( {"Graham", 72} );
```

```
scores.insert( {"Hareesh", 77} );
```

```
for (auto& it: scores)
```

```
    cout << it.first << " has a score of " << it.second << endl;
```

- What is unusual about iterating over unordered containers?
 - Only forward iteration is supported