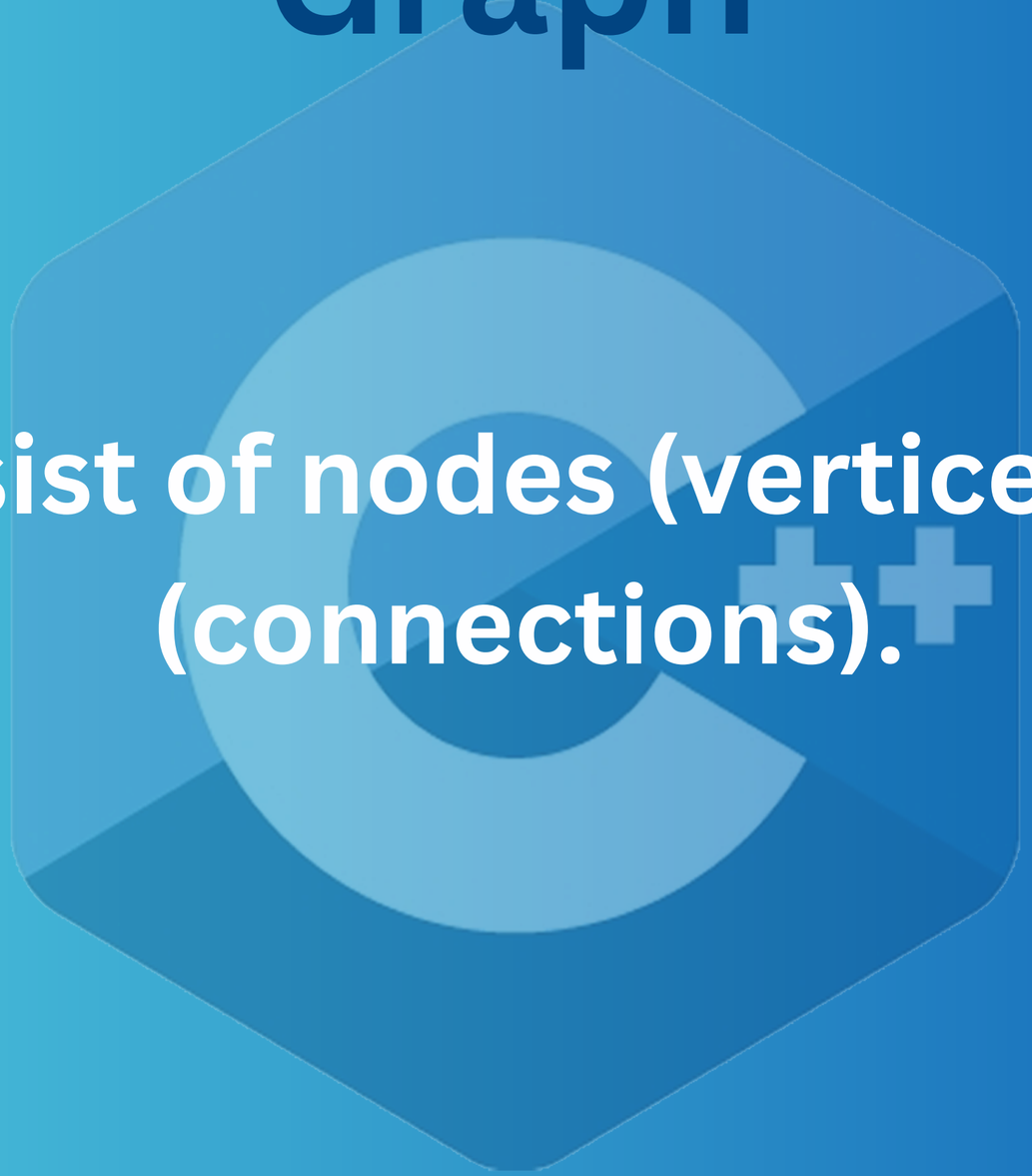
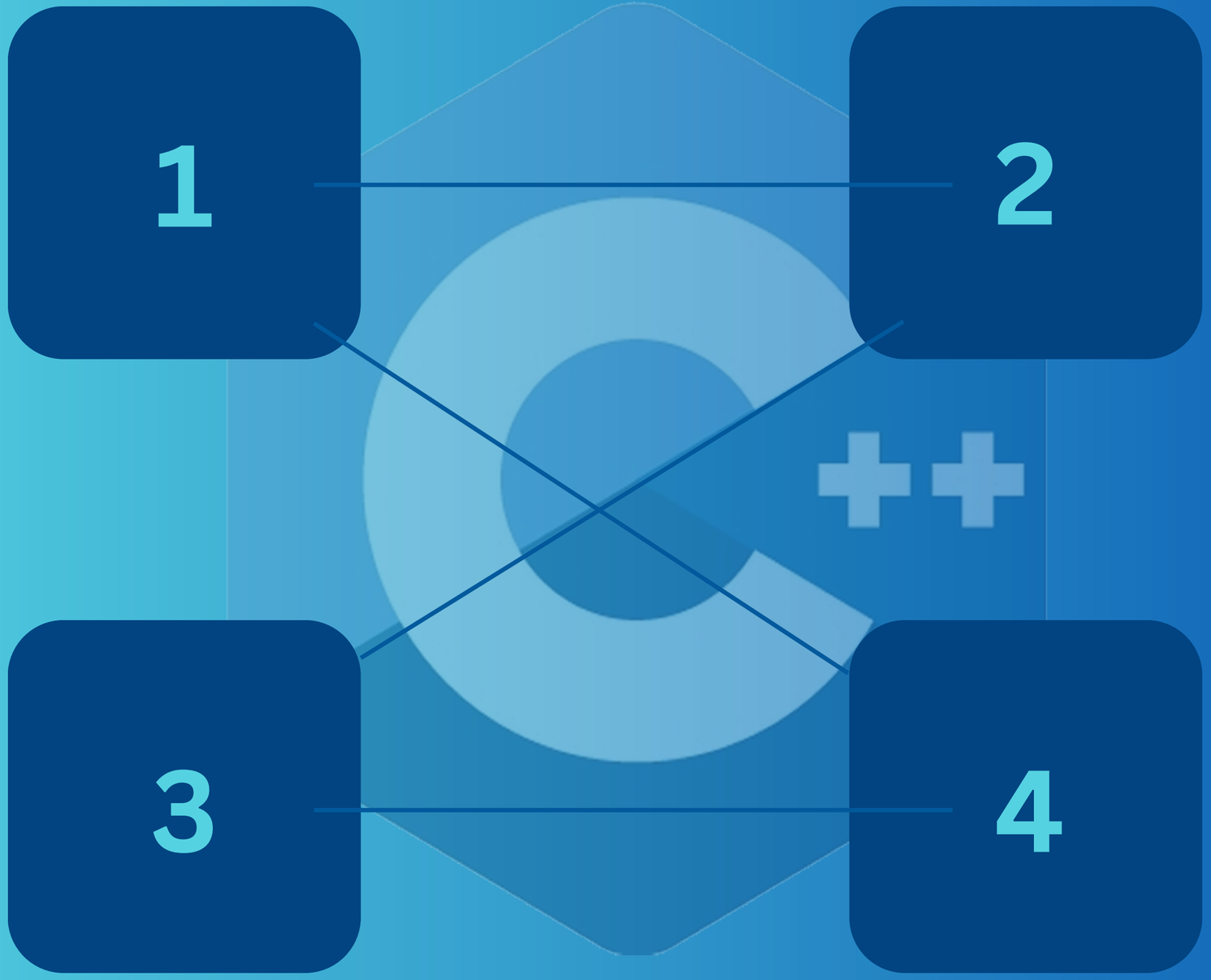
Graph



Graphs consist of nodes (vertices) and edges (connections).


1
2
3
4

graph.cpp U

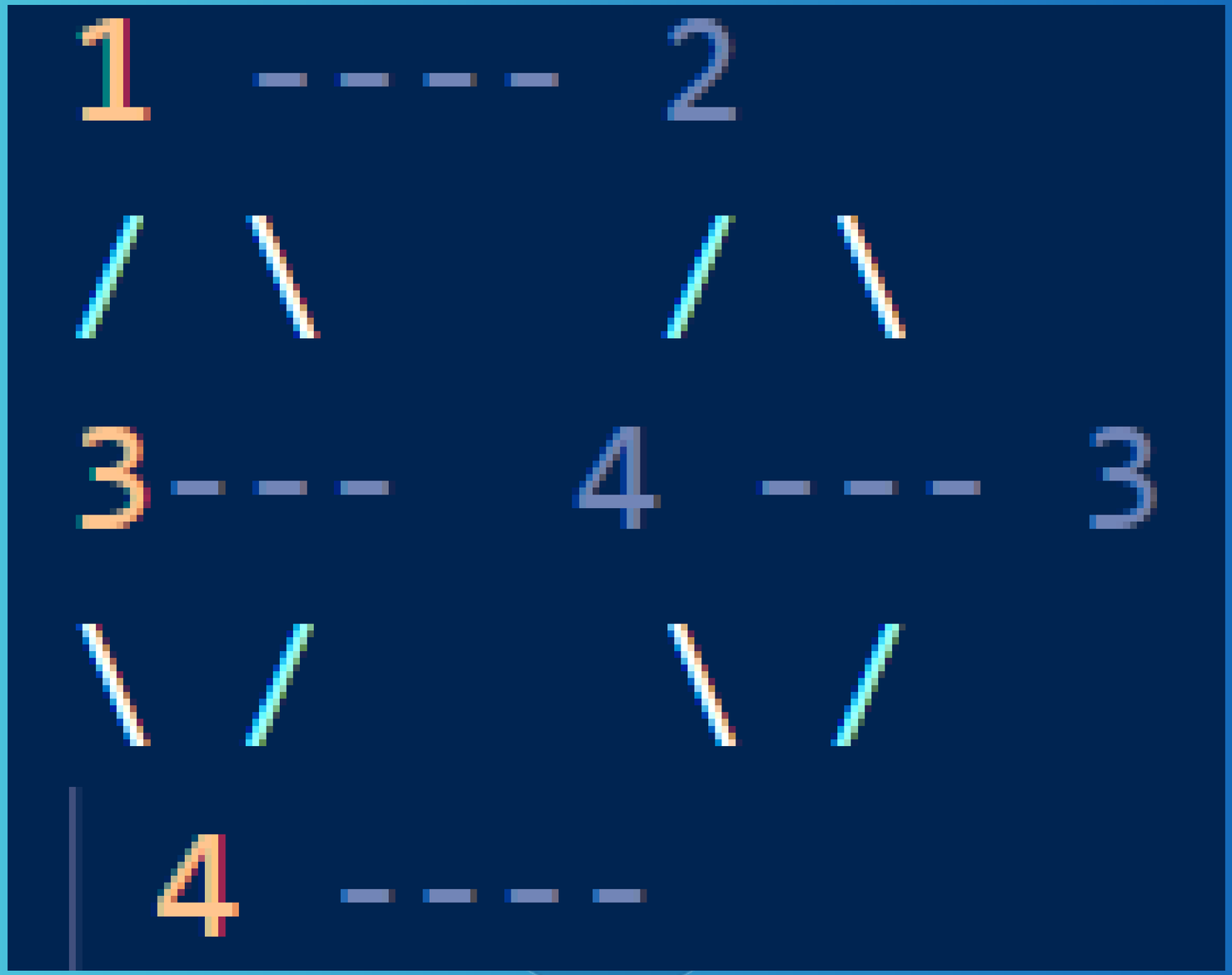
code > graph.cpp > main()

```
26  int main() {
36      Edge e2(1, 3);
37      e2.addEdge(adj, e2);
38      Edge e3(1, 4);
39      e3.addEdge(adj, e3);
40      Edge e4(2, 3);
41      e4.addEdge(adj, e4);
42      Edge e5(2, 4);
43      e5.addEdge(adj, e5);
44      Edge e6(3, 4);
45      e6.addEdge(adj, e6);
46
47      for (int i = 0; i < V; i++) {
48          Node* n = adj[i];
49          cout << i + 1 << " -> ";  // Adjusting index to start from 1 instead of 0
50          while (n) {
51              cout << n->data << " ";
52              n = n->next;
53          }
54          cout << endl;
55      }
56  }
```

PROBLEMS DEBUG CONSOLE OUTPUT PORTS SERIAL MONITOR COMMENTS TERMINAL

powershell - code

```
PS D:\Documents\GitHub\Introduction-to-Data-Structure\code> g++ graph.cpp -o graph
PS D:\Documents\GitHub\Introduction-to-Data-Structure\code> ./graph.exe
1 -> 4 3 2
2 -> 4 3
3 -> 4
4 ->
PS D:\Documents\GitHub\Introduction-to-Data-Structure\code>
```


1 ---- 2
/ \ / \
3--- 4 --- 3
\ / \ /
4 ----

```cpp
#include <iostream>
using namespace std;

struct Node {
    int data;
    Node* next;

    // Constructor
    Node(int v) : data(v), next(NULL) {};
};
```

```
12  struct Edge {
13      // source and destination
14      int src, dest;
15
16      Edge(int s, int d) : src(s), dest(d) {};
17
18      // add
19      void addEdge(Node* adj[], Edge e) {
20          Node* n = new Node(e.dest);
21          n->next = adj[e.src - 1]; // Adjusting index to start from 1 instead of 0
22          adj[e.src - 1] = n;
23      }
24  };
```

```
26  int main() {
27      int V = 4;
28      Node* adj[V];
29
30      for (int i = 0; i < V; i++) {
31          adj[i] = nullptr;
32      }
```

```
34      Edge e1(1, 2);
35      e1.addEdge(adj, e1);
36      Edge e2(1, 3);
37      e2.addEdge(adj, e2);
38      Edge e3(1, 4);
39      e3.addEdge(adj, e3);
40      Edge e4(2, 3);
41      e4.addEdge(adj, e4);
42      Edge e5(2, 4);
43      e5.addEdge(adj, e5);
44      Edge e6(3, 4);
45      e6.addEdge(adj, e6);
```

```
47      for (int i = 0; i < V; i++) {
48          Node* n = adj[i];
49          cout << i + 1 << " -> ";  // Adjusting index to start from 1 instead of 0
50          while (n) {
51              cout << n->data << " ";
52              n = n->next;
53          }
54          cout << endl;
55      }
56  }
```

Next Video

Hashing and Hash Tables