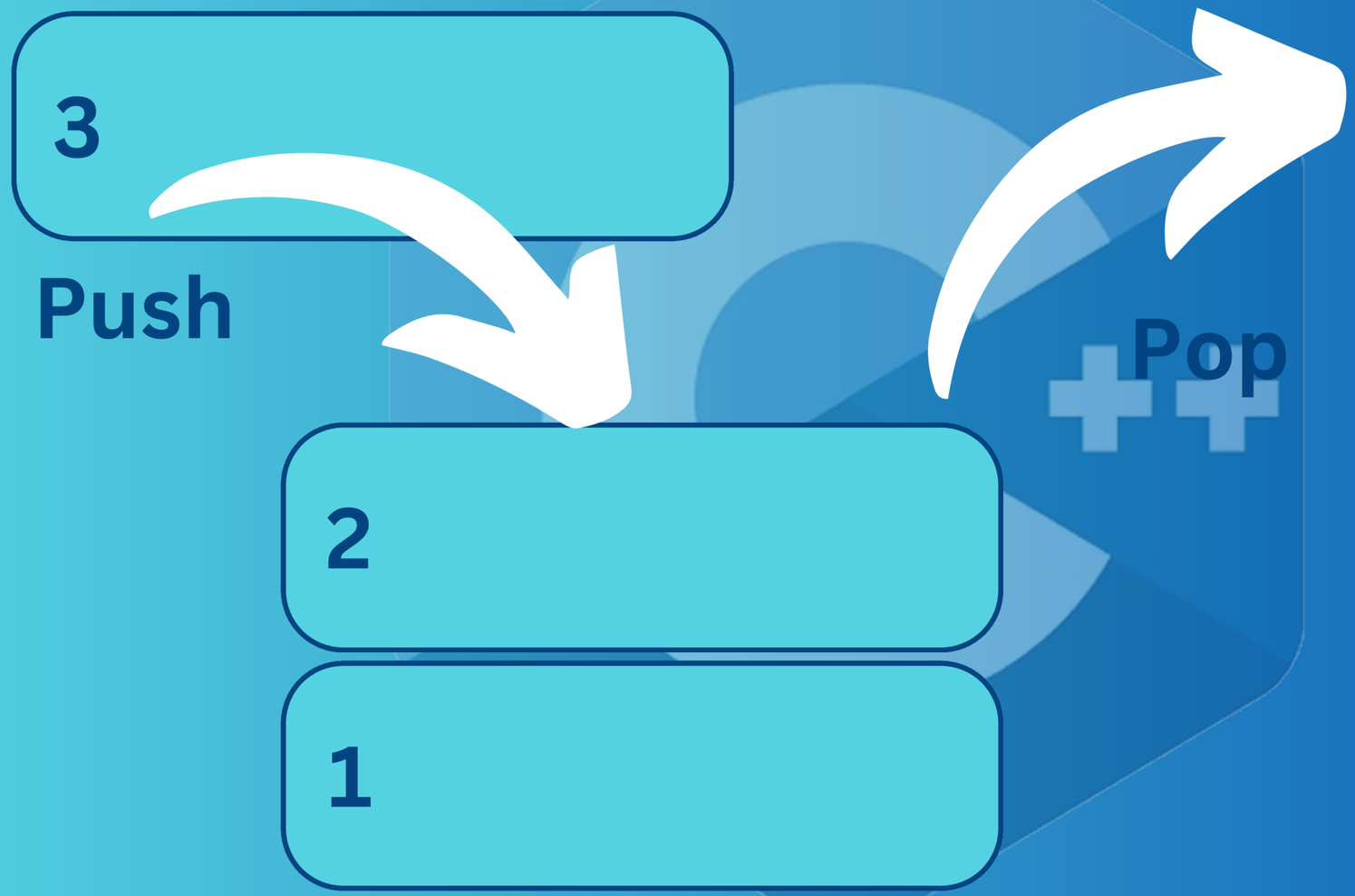
Stack and Queue

Stack is last in first out (LIFO) data structure

Stack



File Edit Selection View Go Run Terminal Help

Introduction-to-Data-Structure

stack.cpp U

code > stack.cpp > main()

```
13  struct Stack{
46
47  int main (){
48      Stack l;
49
50      l.push(1);
51      l.push(2);
52      l.push(3);
53
54      l.print();
55
56      l.pop();
57
58      l.print();
59  }
```

PROBLEMS DEBUG CONSOLE OUTPUT PORTS SERIAL MONITOR COMMENTS TERMINAL

powershell - code

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

main* 0 0 0 Cloud Code - Sign in Ln 58, Col 15 Spaces: 4 UTF-8 CRLF C++ Win32

Code

```
1   // stack is last in first out (LIFO)
2   #include <iostream>
3   using namespace std;
4
5   struct Node{
6       int data;
7       Node* next;
8
9       // constructor
10      Node(int v): data(v) , next (nullptr) {}
11  };
```

```
13   struct Stack{
14       Node* head;
15
16       Stack(): head (nullptr){}
```

```
18       // push will add a new node to the top of the stack
19       void push(int v){
20           Node* n = new Node(v);
21           n -> next = head;
22           head = n;
23       }
```

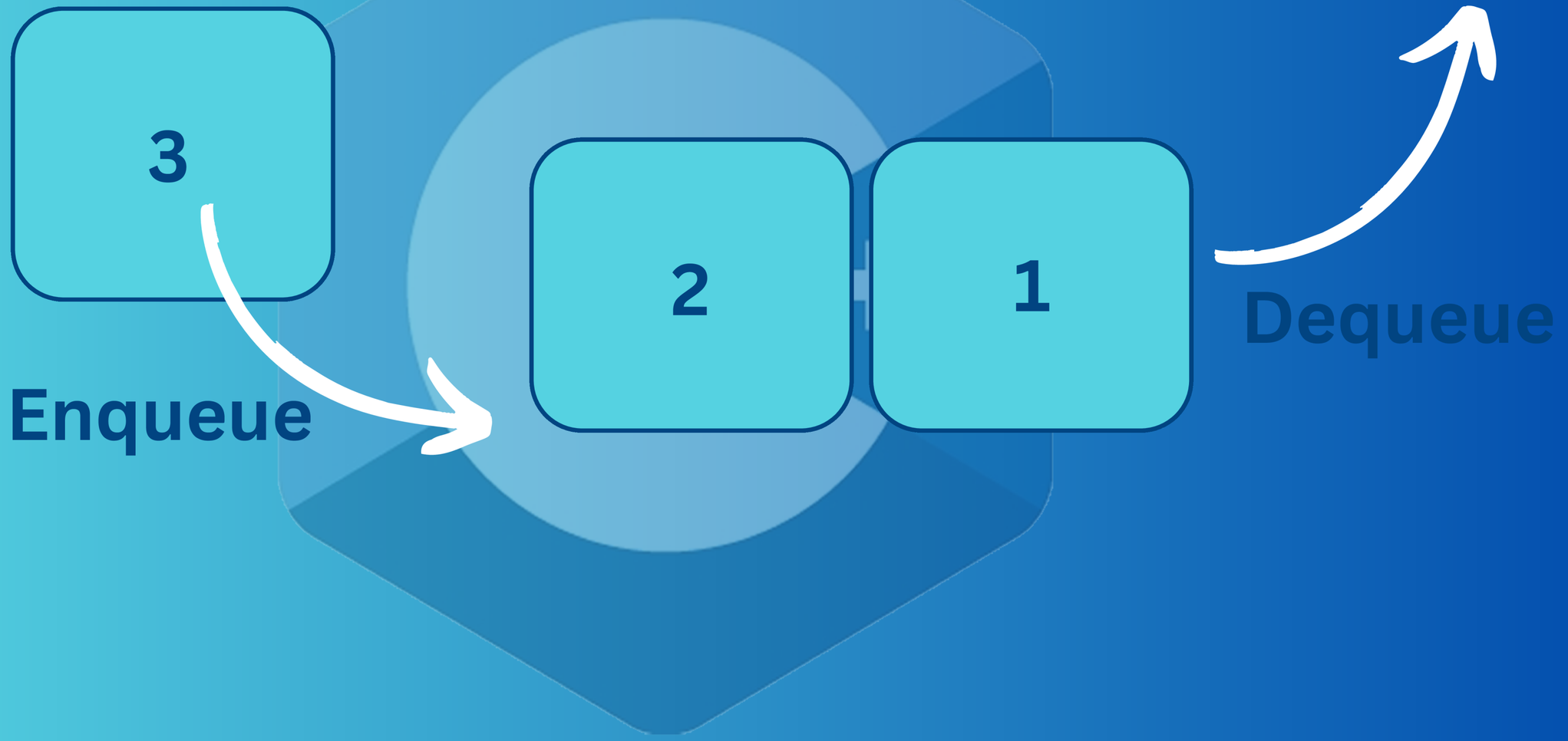
```
25      // pop will remove the top node from the stack
26      void pop(){
27          if(head == nullptr){
28              cout << "stack is empty" << endl;
29          }  else {
30              Node* t = head;
31              head = head -> next;
32              delete t;
33          }
34      }
```

```
36        // print
37        void print(){
38            Node* t = head;
39            while (t != nullptr){
40                cout << t -> data << " -> ";
41                t = t -> next;
42            }
43            cout << "null" << endl;
44        }
45    };
```

Queue

Queue is first in first out (FIFO) data structure

Queue



File Edit Selection View Go Run Terminal Help

Introduction-to-Data-Structure

stack.cpp U | queue.cpp U

code > queue.cpp > main()

```
struct Queue{

int main (){
    Queue l;

    l.enqueue(1);
    l.enqueue(2);
    l.enqueue(3);

    l.print();

    l.dequeue();

    l.print();
}
```

PROBLEMS DEBUG CONSOLE OUTPUT PORTS SERIAL MONITOR COMMENTS TERMINAL

powershell - code

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

main* 0 0 0 Cloud Code - Sign in Ln 64, Col 2 Spaces: 4 UTF-8 CRLF C++ Win32

Code

```cpp
// queue is first in first out (FIFO)
#include <iostream>
using namespace std;

struct Node{
    int data;
    Node* next;

    Node(int v): data(v) , next (nullptr) {}
};
```

```
12    struct Queue{
13        Node* head;
14
15        Queue(): head (nullptr) {}
```

```
17    // enqueue will add a new node to the end of the queue
18    void enqueue(int v){
19        Node* n = new Node(v);
20        if (head == nullptr){
21            head = n;
22        } else {
23            Node* t = head;
24            while (t -> next != nullptr){
25                t = t -> next;
26            }
27            t -> next = n;
28        }
29    }
```

```
31      // dequeue will remove the first node from the queue
32      void dequeue(){
33          if(head == nullptr){
34              cout << "queue is empty" << endl;
35          } else {
36              Node* t = head;
37              head = head -> next;
38              delete t;
39          }
40      }
```

```
42        void print(){
43            Node* t = head;
44            while (t != nullptr){
45                cout << t -> data << " -> ";
46                t = t -> next;
47            }
48            cout << "null" << endl;
49        }
50    };
```

Next Video

Trees and Binary Trees