Structures

A structure is simply a way of grouping related variables together into a single construct. A structure can be passed to a function as a single unit.

An example of a structure is shown below and is used in the Pipes program. It represents a node in the puzzle and contains three variables that hold the coordinates and cell value respectively.

struct Node {

byte x;

byte y;

byte value;

};

Once defined, a new variable can be created using it as a datatype as shown below. The members of the structure can be referenced directly to assign or retrieve their values.

Node aNode;

aNode.x = 3;

aNode.y = 4;

int x = calc(aNode);

int calc(Node theNode) {

return theNode.x \* theNode.y;

}

The declaration and initialization code above could have been expressed in a single line as shown below. The members of the structure must be initialised in the sequence they were declared in the structure definition. Not all members must be initialized and the second example is also valid and the *value* member would be initialised to 0.

Node myNewNode = {3, 4, 5};

Node myNewNode = {3, 4};

Structures can contain other structures. The Pipes game contains a simple structure for the player’s game state including the selected node and highlighted node as shown below.

struct Player {

Node highlightedNode;

Node selectedNode;

}

Initialising a structure with embedded structures is shown below. Note that the initializer for each node is contained within its own braces {}.

Player player = {

{3, 4, 5}, // init highlighted node

{3, 4 } // init selected node

};