Component 1: Data Organisation

Two-dimensional array example

	[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]
[0]	37	11	42	6	26	56	4	76
[1]	98	203	64	23	126	79	14	23
[2]	30	1	4	13	29	48	21	211
[3]	10	57	73	110	82	29	289	245

- 32 elements
- Elements are indexed by two numbers, one for its row and one for its column `[y,x]`
- Each element can be accessed using its index
- The element at index `[1,7]` is 23.

Key terminology

Term	Definition
Validation	Ensures that data entered is reasonable.
Verification	Ensures that data entered is consistent.

Validation

Presence checks

Used to check if a required field is left blank.

```
1 if dataEntered = "" then
2     output error message
3 end if
```

Format checks

Used to ensure data matches a specific pattern, such as the format, or pattern, of a postcode.

```
1 if postcode <> format(LL00 0LL) then
2     output error message
3 end if
```

Length checks

Used to ensure an input data string is a sensible length, such as the number of digits in a phone number should be 11.

```
1 if len(telNo) <> 11 then
2     output error message
3 end if
```

Type checks

Used to ensure input data is a particular data type, e.g. quantity ordered to be integer or cost to be real.

Range checks

Used to ensure input data lies within a specified range, such as overtime hours to be > 0 and < 15.

```
1 if hours < 0 OR hours > 15 then
2     output error message
3 end if
```

Verification

Double-entry

Double entry involves comparing two versions of data input, such as "re-enter your password".

```
1 if password <> reTypePassword then
2     output error message
3 end if
```

A verification algorithm will compare the two versions and inform the user if they are not identical.

Screen based / visual check

Requires the user to check a display of input data and confirm that it is correct.

Check digit

More sophisticated verification algorithms apply calculations to input data, e.g. to produce the check digits of bar codes. Repeating the calculations and checking the result is the same can verify the data.