Validation, Verification and Testing

Validation

Validation enables programs to check that data entered meets certain rules.

Types of validation include:

- Range Checks
- Length Checks
- Type Checks
- Check Digits

Range Checks

Range checks are used to test whether a value is within a specified range.

This example pseudocode checks whether the mark entered is in the range 0 to 25.

```
Mark ← INPUT
IF Mark >= 0 AND Mark <= 25
    THEN
        OUTPUT "Mark valid"
    ELSE
         OUTPUT "Mark out of range"
ENDIF
```

Input	5
Output	

Range Checks

Range checks are used to test whether a value is within a specified range.

This example pseudocode checks whether the mark entered is in the range 0 to 25.

```
Mark ← INPUT
IF Mark >= 0 AND Mark <= 25
    THEN
        OUTPUT "Mark valid"
    ELSE
        OUTPUT "Mark out of range"
ENDIF
```

Input	5
Output	“Mark Valid”

Length Checks

Length checks are used to test whether the value entered is of a specified length.

This example pseudocode checks whether the password entered is at least 5 characters in length.

```
Password ← INPUT
IF LEN(Password) < 6
    THEN
        OUTPUT "Password too short"
    ELSE
        OUTPUT "Password valid"
ENDIF
```

Input	“Hello”
Output	

Length Checks

Length checks are used to test whether the value entered is of a specified length.

This example pseudocode checks whether the password entered is at least 5 characters in length.

```
Password ← INPUT
IF LEN(Password) < 6
    THEN
        OUTPUT "Password too short"
    ELSE
        OUTPUT "Password valid"
ENDIF
```

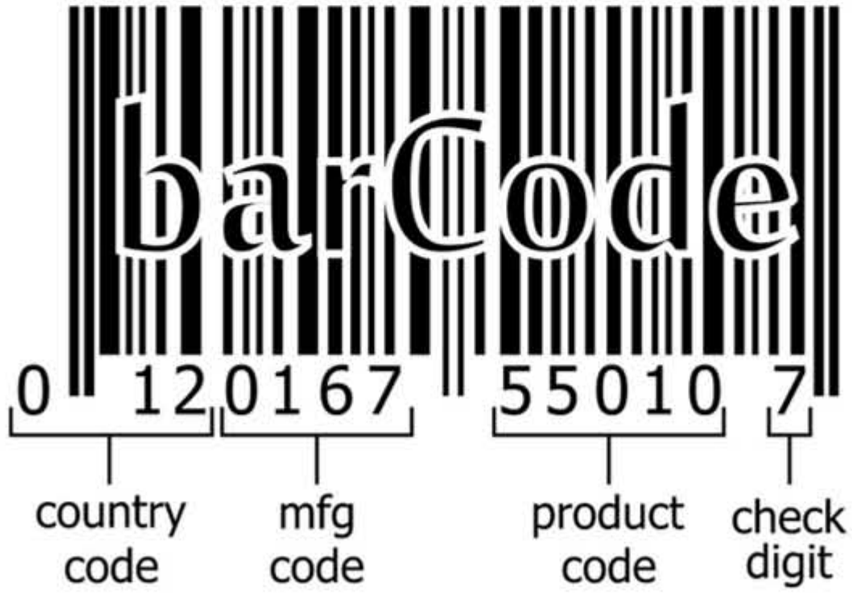
Input	“Hello”
Output	“Password too short”

Type Checks and Check Digits

Type checks test the inputted data to ensure it matches the required data type. For example, a score might be checked to make sure it is an integer.

A check digit is a digit added to the end of an identification number that is used to test whether the number has been entered correctly.

Check digits are widely used in product barcodes.



Test Data

It is important to test the validation that we build into our programs to ensure it works as we expect it to.

There are three types of test data that it is standard to use when testing algorithms and programs.

Normal	Boundary	Erroneous
Data that is well within the normal range of what should be accepted.	Data that is at the upper or lower range of what should be accepted.	Data that is invalid and should not be accepted.

Sometimes data can be both boundary and erroneous if it is just outside the range of what should be accepted.

Test Table Example

```
Mark ← 25
IF Mark >= 0 AND Mark <= 25
        THEN
                OUTPUT "Mark valid"
        ELSE
                OUTPUT "Mark out of range"
ENDIF
```

Description	Test Type	Test Data	Expected Outcome	Result
Test the highest allowable integer	Boundary	25	‘Mark valid’	‘Mark valid’
Test value higher than allowable range	Erroneous	30	‘Mark out of range’	
Test Value within allowable range	Normal	10	‘Mark valid’	

Test Table Example

```
Mark ← 30
IF Mark >= 0 AND Mark <= 25
        THEN
                OUTPUT "Mark valid"
        ELSE
                OUTPUT "Mark out of range"
ENDIF
```

Description	Test Type	Test Data	Expected Outcome	Result
Test the highest allowable integer	Boundary	25	‘Mark valid’	‘Mark valid’
Test value higher than allowable range	Erroneous	30	‘Mark out of range’	‘Mark out of range’
Test Value within allowable range	Normal	10	‘Mark valid’	

Test Table Example

```
Mark ← 10
IF Mark >= 0 AND Mark <= 25
        THEN
                OUTPUT "Mark valid"
        ELSE
                OUTPUT "Mark out of range"
ENDIF
```

Description	Test Type	Test Data	Expected Outcome	Result
Test the highest allowable integer	Boundary	25	‘Mark valid’	‘Mark valid’
Test value higher than allowable range	Erroneous	30	‘Mark out of range’	‘Mark out of range’
Test Value within allowable range	Normal	10	‘Mark valid’	‘Mark valid’

Verification

Verification checks user input by asking the user to input the same data more than once to verify it is correct.

The pseudocode below verifies the password the user has entered.

```
Password ← INPUT
PasswordVerify ← INPUT
IF Password = PasswordVerify
    THEN
        OUTPUT "Passwords match"
    ELSE
        OUTPUT "Password do not match"
ENDIF
```