

# Programming Techniques

Keyword	Definition
<b>Procedure</b>	A subroutine with no value returned
<b>Function</b>	A subroutine which does return a value after processing has taken place
<b>Parameter</b>	A value which is passed to a procedure or function
<b>Scope</b>	The part of the program in which a variable or constant is recognised and can be used
<b>Syntax error</b>	The code written doesn't conform to the rules of the language
<b>Logical error</b>	The program will run, but it doesn't work as the programmer intended
<b>Normal data</b>	Checks the data in the valid range
<b>Boundary data</b>	Checks the values either side of the boundary
<b>Erroneous data</b>	Checks that data of the wrong type or invalid data is rejected

## Procedures

```

SUBROUTINE showKeys()
  OUTPUT "Keyboard controls"
  OUTPUT "===== "
  OUTPUT "1:  Up arrow - forwards"
  OUTPUT "2:  Down arrow - backwards/brake"
  OUTPUT "3:  Left arrow - turn left"
  OUTPUT "4:  Right arrow - turn right"
  OUTPUT "5:  R - refuel"

ENDSUBROUTINE

showKeys()
    
```

## Functions

```

SUBROUTINE sum(a, b)
  total ← a + b
  return total
ENDSUBROUTINE

answer ← sum(5, 3)
OUTPUT answer
    
```

This is a function as it returns a value

5 and 3 are parameters being passed to the function

## Trace Tables

```

num ← 3
n ← 0
WHILE n < 4
  num ← num + n
  n ← n + 1
ENDWHILE
OUTPUT num
    
```

num	n	n < 4	OUTPUT
3	0	TRUE	
3	1	TRUE	
4	2	TRUE	
6	3	TRUE	
9	4	FALSE	9

## Variable scope

- Variables **var1** and **var2** are global variables and can be seen anywhere in the program
- Variables **a**, **b** and **c** can only be seen and used inside **s1**
- Variables **x**, **y** and **z** can only be seen and used inside **s2**

Variables var1, var2

```

SUBROUTINE s1()
  variables a, b, c
    
```

```

SUBROUTINE s2()
  variables x, y, z
    
```

## Validation checks

Check	Example
Range check	A number or date is within a sensible/allowed range
Type check	Data is of the right type, such as integer, letter or text
Length check	Text entered is not too long or too short – for example, a password is between 8 and 15 characters
Presence check	Checks that data has been entered, i.e. the field has not been left blank
Format check	Checks that the format of, for example, a postcode or email address is correct

# Programming Techniques

## Errors

```
SUBROUTINE max(a,b,c)
  IF a ≥ b AND a > c
    return a
  ELSE IF b ≥ a AD b ≥ c
    return b
  ELSE
    return a
  ENDIF
ENDSUBROUTINE

OUTPUT "Enter number: "
num1 ← USERINPUT
OUTPUT "Enter number: "
num2 ← USERINPUT
OUTPUT "Enter number: "
num3 ← USERINPUT

maxNum ← max(num1, nm2, nu3)

OUTPUT "maximum number is: "
OUTPUT mxNum
```

There are six mistakes in this code


## Logical Errors

```
SUBROUTINE max(a,b,c)
  IF a ≥ b AND a > c
    return a
  ELSE IF b ≥ a AD b ≥ c
    return b
  ELSE
    return a
  ENDIF
ENDSUBROUTINE

OUTPUT "Enter number: "
num1 ← USERINPUT
OUTPUT "Enter number: "
num2 ← USERINPUT
OUTPUT "Enter number: "
num3 ← USERINPUT

maxNum ← max(num1, nm2, nu3)

OUTPUT "maximum number is: "
OUTPUT mxNum
```



The program will run, but it won't work as the programmer intended

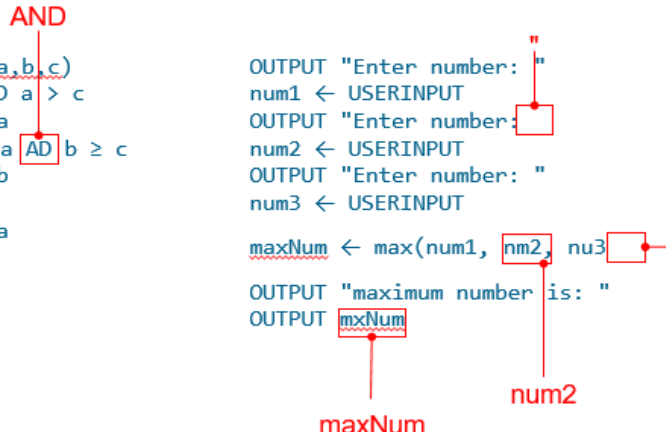
## Syntax Errors

```
SUBROUTINE max(a,b,c)
  IF a ≥ b AND a > c
    return a
  ELSE IF b ≥ a AD b ≥ c
    return b
  ELSE
    return a
  ENDIF
ENDSUBROUTINE

OUTPUT "Enter number: "
num1 ← USERINPUT
OUTPUT "Enter number: "
num2 ← USERINPUT
OUTPUT "Enter number: "
num3 ← USERINPUT

maxNum ← max(num1, nm2, nu3)

OUTPUT "maximum number is: "
OUTPUT mxNum
```



The rules of the programming language have been broken

## Test Data

**“Enter a number between 1 and 100”**

Some examples of test data:

- **Normal data:** 5 (checks a single digit), 14 (checks two digits)
- **Boundary data:** checks the values either side of the boundary. 1 (checks lowest valid data), 100 (checks highest valid data), 0 (checks invalid data at the boundary), 101 (checks invalid data at the boundary).
- **Erroneous:** -5 (checks a negative number), 1012 (checks a large number), "\$#" (checks that data of the wrong type or that is invalid is rejected)