Reference language for Computing Science question papers (summary)

This document summarises the reference language used to present code in SQA Computing Science question papers for National 5, Higher and Advanced Higher qualifications.

This edition: September 2016, version 1.0

Published by the Scottish Qualifications Authority
The Optima Building, 58 Robertson Street, Glasgow G2 8DQ
Lowden, 24 Wester Shawfair, Dalkeith, Midlothian EH22 1FD

www.sqa.org.uk

Contents

National 5 reference language

Questions assessing understanding and application of programming skills will (mainly) be presented using SQA’s standardised reference language, which may include the following terms:

Base types:	INTEGER, REAL, BOOLEAN, CHARACTER
Structured types:	STRING
	ARRAY OF ..
Structured values:	" .. ", [..], { .. }, id(..)
System entities:	DISPLAY, KEYBOARD
Variable introduction:	DECLARE .. INITIALLY
	DECLARE .. AS .. INITIALLY
Assignment:	SET .. TO ..
Conditions:	IF .. THEN .. END IF
	IF .. THEN .. ELSE .. END IF
Conditional repetition:	WHILE .. DO .. END WHILE
	REPEAT .. UNTIL ..
Fixed repetition:	REPEAT .. TIMES .. END REPEAT
Iteration:	FOR .. FROM .. TO .. DO .. END FOR
	FOR .. FROM .. TO .. DO .. STEP .. END FOR
	FOR EACH .. FROM .. DO .. END FOR EACH
Input / output:	RECEIVE .. FROM ..
	DECLARE .. AS .. INITIALLY FROM ..
	SEND .. TO ..
Operations:	-, +, *, /, ^, MOD, &
Comparisons:	=, ≠, <, ≤,, >, ≥
Logical operators:	AND, OR, NOT
Subprograms:	id(parameters)

< .. > is used to indicate an *elision* — a code fragment expressed in English, not in the formal reference language
# is used to indicate comments

Higher reference language

Questions assessing understanding and application of programming skills will (mainly) be presented using SQA’s standardised reference language, which may include the following terms:

Base types:	INTEGER, REAL, BOOLEAN, CHARACTER
Structured types:	STRING
	ARRAY OF ..
	RECORD .. IS { .. }
Structured values:	" .. ", [..], { .. }, id(..)
System entities:	DISPLAY, KEYBOARD
Variable introduction:	DECLARE .. INITIALLY
	DECLARE .. AS .. INITIALLY
Assignment:	SET .. TO ..
Conditions:	IF .. THEN .. END IF
	IF .. THEN .. ELSE .. END IF
Conditional repetition:	WHILE .. DO .. END WHILE
	REPEAT .. UNTIL ..
Fixed repetition:	REPEAT .. TIMES .. END REPEAT
Iteration:	FOR .. FROM .. TO .. DO .. END FOR
	FOR .. FROM .. TO .. DO .. STEP .. END FOR
	FOR EACH .. FROM .. DO .. END FOR EACH
Input / output:	RECEIVE .. FROM ..
(including files)	DECLARE .. AS .. INITIALLY FROM ..
	SEND .. TO ..
File Operations:	OPEN ..
	CLOSE ..
	CREATE ..
Operations:	-, +, *, /, ^, MOD, &
Comparisons:	=, ≠, <, ≤,, >, ≥
Logical operators:	AND, OR, NOT
Subprograms:	id(parameters)

Where required, subprograms may be presented in the following formats:

```
PROCEDURE id ( parameters )
        commands
END PROCEDURE

FUNCTION id( parameters ) RETURNS type
        commands
        RETURN expression
END FUNCTION
```

< .. > is used to indicate an *elision* — a code fragment expressed in English, not in the formal reference language
# is used to indicate comments

Advanced Higher reference language

Questions assessing understanding and application of programming skills will (mainly) be presented using SQA's standardised reference language, which may include the following terms:

Base types:	INTEGER, REAL, BOOLEAN, CHARACTER
Structured types:	STRING
	ARRAY OF ..
	RECORD .. IS { .. }
	CLASS .. IS { .. } METHODS ... END CLASS
	CLASS .. INHERITS .. WITH { .. } METHODS .. END CLASS
	CONSTRUCTOR .. END CONSTRUCTOR
	OVERRIDE CONSTRUCTOR .. END CONSTRUCTOR
Structured values:	" .. ", [..], { .. }, id(..)
System entities:	DISPLAY, KEYBOARD
Variable introduction:	DECLARE .. INITIALLY
	DECLARE .. AS .. INITIALLY
Assignment:	SET .. TO ..
Conditions:	IF .. THEN .. END IF
	IF .. THEN .. ELSE .. END IF
Conditional repetition:	WHILE .. DO .. END WHILE
	REPEAT .. UNTIL ..
Fixed repetition:	REPEAT .. TIMES .. END REPEAT
Iteration:	FOR .. FROM .. TO .. DO .. END FOR
	FOR .. FROM .. TO .. DO .. STEP .. END FOR
	FOR EACH .. FROM .. DO .. END FOR EACH
Input / output:	RECEIVE .. FROM ..
(including files)	DECLARE .. AS .. INITIALLY FROM ..
	SEND .. TO ..
File Operations:	OPEN ..
	CLOSE ..
	CREATE ..
Operations:	-, +, *, /, ^, MOD, &
Comparisons:	=, ≠, <, ≤,, >, ≥
Logical operators:	AND, OR, NOT
Subprograms:	id(parameters)

Where required, subprograms may be presented in the following formats:

```
PROCEDURE id ( parameters )
        commands
END PROCEDURE

FUNCTION id( parameters ) RETURNS type
        commands
        RETURN expression
END FUNCTION
```

< .. > is used to indicate an *elision* — a code fragment expressed in English, not in the formal reference language
# is used to indicate comments