PseudoCode Cheat Sheet

by mason via cheatography.com/35063/cs/11011/

Data Types

STRING

INTEGER

REAL

BOOLEAN

CHARACTER

Arithmetic Operators

Addition: +	Division: /
Subtraction: –	Remainder: MOD
Multiplication: *	Integer Division: DIV

Other

Assignment operator: = Don't confuse = with == (they're different)

You can use true and false to check Boolean variables in your conditions (as in IF SomeBooleanVariable == true THEN … etc.)

Variables are not loosely typed – you can't mix different types of variable They don't seem to have to be declared, although sometimes there'll be a declaration (If you're asked to declare something, you can pretty much just make it up – as long as it specifies the data type, identifier, etc…)

Sometimes a colon is used to identify the data type of a variable, e.g. SomeVariable : REAL would declare a real (decimal) variable

Keywords are in capitals in pseudocode

Arrays work as they do in most languages, but often their index starts at 1, rather than 0, and sometimes they use parenthesis ( ) instead of brackets [ ]

Multidimensional arrays work like this: identifier(y, x)

Constructs

```
IF condition THEN
       do something
ELSE
       do something else
END IF
WHILE condition
        sta tements
END WHILE
REPEAT
        sta tements
UNTIL condition
```

The else bit is optional – you don't have to have it

The condition is any expression that evaluates to a Boolean

Notice how the statements are indented (well, hopefully they'll display indented when I publish this…)

When using WHILE and REPEAT loops, to count, you need to manually initialize and increment the counting variable

The statements inside a loop should cause a change in one of the values in the condition, otherwise you may create an infinite loop

Procedure & Functions

```
PROCEDURE doSomething(Parameter
: DATATYPE, OtherParameter:
DATATYPE)
        sta tements
END PROCEDURE
FUNCTION doSome thi ng( par -
ameter: DATATYPE) : RETURNTYPE
        sta tements
RETURN something
END PROCEDURE
```

Procedures and functions don't have to take parameters, but the parentheses () are necessary

Functions must have a return type and must return something (of that type)

File Handling

For

```
FOR i = 1 to 10
        sta tements
NEXT i
END FOR
```

With a FOR loop, incrementing and initializing of the counting variable are done automatically

You can call the counting variable (i, in this case) anything you want, and you can also set the = something TO something values to whatever you want it to count from and to

Relational / Comparison Operators

Equal to: ==	Not equal to: !=
Not equal to: <>	Greater than: >
Less than: <	Greater than or equal to: >=
Less than or equal to: <=	Both: AND
One or both: OR	Invert: NOT

Logical / Boolean Operators

CASE

```
CASE OF something
        som ething = this:
statement
        som ething = that:
statement
        som ething = other:
statement
        def ault: statement
END CASE
```

In a case statement (that's switch statement to Java/C# people) you can have however many options you want, but there must always be a default for if none of the options match

In the example above, something is the variable that is being checked, and this, that and other are things it's being compared with

String Manipulation

There are two functions that look things up in the ASCII character set table for you:

ASCII(character) returns the ASCII value of a character, character

CHAR(integer) returns the character of an ASCII value, integer

Characters may be in single or double quotes (it’s another thing the examiners don’t seem to have made their minds up about)

Strings can be concatenated using the addition operator, +

You sometimes have to use concatenation to output something in a friendly way

Mathematics can’t be done on strings, but you can compare their ASCII values using the relational operators (<, >, <>, !=, ==, >=, <=)

MID(string, integer1, integer2) returns the part of the string between positions integer1 and integer2

LEFT(string, integer) RIGHT(string, integer) LENGTH(string) returns the length of the string, string

LOCATE(string1, string2) returns the position of the first occurrence of string2 in string1 (0 means it starts at the beginning, -1 means it’s not in there)

I don’t know why these functions are acting like there are official pseudocode libraries… they must just be for inspiration, and as a guide

File Handling

OPEN filename FOR MODE	(You can open as READ or WRITE only, one at a time)
READ extracted-variable FROM filename	WRITE something TO filename
CLOSE filename	DELETE filename
RENAME filename TO something	CREATE filename

“filename is at end of file” can be used in the condition of loops, to iterate through all the records

High-Level Questions

Occasionally, you’ll be asked to write something in a real language
I think the course does require you to be taught the basics of a high-level procedural programming language in the first year
Again, though, the questions are more about understanding what to do than following the correct syntax
Familiarise yourself with how to write the constructs above in a high-level language (i.e. not assembly…)