Higher Computing Science

Modular –Summary Notes

Modular Code

Modular code is code that uses procedures and functions.

Sub Program	Function
A section of code that performs a specific task.	A section of code that returns a value.
Example	Example

CALL display_results(average)
 	
Sub display_results(byval average as single)
 	
Lstresults.items.add("The average mark is "& average)
 	
End Sub
	grade = decide_grade(percentage)
	
	Function decide_grade(byval percentage as integer)
	If percentage >= 75 then
	grade="A"
	elseif percentage >= 60 then
	grade = "B"
	elseif percentage >= 50 then
	grage = "C"
	elseif percentage >= 40 then
	grade = "D"
	else
	grade = "Fail"
	end if
	Return grade
	End function

NOTE **Sub programs are called using the CALL command**	NOTE **Functions are called using assignment e.g. grade = decide_grade(percentage)** **This is because a function always returns a value.**

Benefits of Modular Code

- Projects can be split up between different teams of programmers. Each team can be working on code simultaneously
- Sub-programs make code easier to read
- If code is readable it is easier to maintain
- If code is readable it is easier to spot errors

Parameters

Parameters are variables that are passed in and out of sub programs and functions. We pass parameters using two different methods

Pass by Reference	Pass by Value
• **The memory address of the variable is passed to the sub-program** • **Any changes made to the variable are passed back out to be used further on in the code** • **Pass by reference is used when a sub-program is required to alter/update a variable**	• A copy of the value of the variable is passed to the subprogram • Any changes made to the data do not affect the value of the original variable • Pass by value is used when the sub-program needs access to a variable but will not make changes to it

Arrays are always passed by reference. This is because if we passed arrays by value the program would create a second copy. Arrays are often large data structures so this is

- Inefficient use of processor time to make the second copy
- Inefficient use of RAM to store the second copy

Actual & Formal Parameters

- Formal parameters are used when the subroutine is declared
- Actual parameters are used when the subroutine is called

Functions

There are two types of functions.

- **User defined functions**
  - These are written by the programmer to perform a specific task
- **Pre-defined functions**
  - These have been pre-written and are built into the programming language for programmers to use as required
- **Advantages of pre-defined functions**
  - Saves the programmer lots of time, this is because
    - the code has already been written, therefore programmers have less code to create

- the code has already been tested, therefore less time spent debugging

Pre-Defined Functions for N5

Function	Purpose	Example	Description
ROUND	Round a numeric value to a specified number of decimal places	Math.round(average, 0)	Rounds average to 0 decimal places
		Math.round(weight, 2)	Rounds weight to 2 decimal places
RANDOM	Generates a random number	Randomize RandomNumber = Int (RND * 10)	Generates a random number between 0 and 9
		Randomize RandomNumber = Int (RND * 9) + 1	Generates a random number between 1 and 10
		Randomize RandomNumber = Int(Rnd * 50)	Generates a random number between 0 and 50
LENGTH	Returns the length of a string	LEN("Jessica") returns 7	Returns the number of characters the string Checks if the number of characters in ID **IS NOT** equal to 5
		If LEN(ID) <> 5	

Pre-Defined Functions for Higher

Function	Purpose	Example	Description	Notes
MID$	Substring	Mid$(***string, start, number of characters***)	Extracts a substring from a string	
		Mid$("Word", 2, 3) returns ord		
		Mid$("computer, 4, 3) returns put		
ASC	ASCII	Asc("A") returns 65	Returns the ASCII value of a character	**A-Z (upper case)** ASCII values between 65 and 90 **a – z(lower case)** ASCII values between 97 and 122
CHARACTER	CHR	Chr(97) returns a	Takes an ASCII value and returns the corresponding character	
INTEGER	INT	INT(3.7556) returns 3	Returns the whole number part of a real number	
CONVERSION INTEGER	CINT	CINT(3.7556) returns 4	Rounds a real number to the nearest integer	
MODULUS	MOD	First MOD second	Returns the remainder of first divided by second	
		7 MOD 3 returns 1		

Other Useful Functions

Function	Purpose	Example	Description
LCASE	Converts a string to lower case	Name = LCASE(JENNIFER)	Name would now store jennifer
UCASE	Converts a string to upper case	Name = UCASE(Jennifer)	Name would now store JENNIFER
SQRT	Returns the square root of a numeric value	Number = SQRT(49)	Number would store the value 7