Concepts and Principles

Sequence

Sequence is the order in which the actions and events are executed.

Sequences are built up from statements including procedure calls, expressions and etc.

No lines of code can be skipped in sequence, it will run and perform every command in order.

An example of Sequence is when we perform hand execution; we read and write the results and variables in order which is the same as in sequence so we can work out what is going wrong and how the program is working out its values.

Repetition

Repetition has conditions which if not met will repeat execution of a block of statements.

If the condition is true it will repeat the procedures tasks until the condition is false.

Iteration is another name for repetition which is why most for loops will be run with a variable called i.

A good example of repetition use is using a for loop to repeat lines of code to perform actions to the array's values. Another example would be using a repeat until loop to run code until the condition is met.

Selection

A question is asked and depending on the answer the program could take multiple paths to different courses of action.

Selection uses statements such as if, else if or else to direct which path to take decided through the conditions set.

An example would be to use an if statement to determine if a user is over 18 to join a website.

Modularisation

Modularity is a technique used by programmers to separate functionality of a program to separate parts. This is done for easy debugging as you can make a module, test it and then use it if nothing goes wrong. It also makes it very easy to remove the module if it doesn't function the way it was designed.

Examples of modularity include making a new procedure for different features of the program such as having a procedure to just update the game or draw the screen.

Programming Artefacts

Function

Function is the way to capture all the steps needed to calculate a value.

Functions are a sequence of tasks used to calculate values. Functions return a result value when the function finishes. Functions can only be used in expressions. Functions have identifiers so you can function call, you can pass parameters and store local variables in the function.

An example of functions would be to randomly generate data and have it returned or to collect information from a user to be returned to a procedure.

Procedure

A group of instructions to perform a task built up from statements which tell the computer what to do.

Procedures are similar to Functions but they do not return a value but instead perform certain tasks. Procedures have Parameters to pass values, have an identifier to name the procedure for procedure call and can be stored in libraries to be called upon.

An example would be anything from sorting values in a array to printing highscores from an array.

Variable

A variable is used to store data in programs. The variable is a container for the value not actually the value itself. The value in the variable can be read and changed at any time.

To assign a value to the variable you must use an assignment statement.

Variables are identified with a name by identifiers so it can be easily referenced and called when required.

Variables are have their own type which allocates space for the variable.

An example would be to use a variable for keeping the users age and name.

Constant

Similar to variable but contains a value that will not change.

Constants are Perfect for values that will not change such as gaps between blocks or amount of columns or rows in a grid.

Type

A type is different ways of a computer storing and interpreting variables.

The type indicates how the data will be stored in the computers memory and how it will be interpreted in the program.

Types examples

Boolean: true or false

String: text characters

Single: number 4 bytes 7-8 digits long

Double:number 8 bytes 15-16 digits long

Char: holds one character aka ‘C’

Integer: whole numbers

If Statement

Alter a sequence of code and respond to different conditions.

The code in an if statement will only execute if the conditions are met.

The if branching statement is useful for using for 2 or more paths.

An IF statement is a Pre-test loop which means it checks the condition before running the code inside so it can run 0 or more times.

An example would be checking the age of a player and if less than 18 it wold tell them to get out, if over 18 would tell them to enter and otherwise tell them to renter a value.

Case Statement

Case statement is a branching code that works by evaluating expressions and matches it to one of a number of different paths.

Case statement is good for handling many alternative paths

Case statements are good for menus.

Array

An array is similar to a variable as it has a type, identifier and requires to be assigned.

An array is a way of storing multiple values inside one group.

Great for storing multiple values such as names, ages, gender and more.

For Loop

For loop is a pre-test loop that repeats a block of code a number of times.

For loops are good for counting values through an array

Examples of for test loops include using it to search all the values through the array so your can sort them in a specific order.

While loop

Continues running until condition is met.

A while loop is a Pre-test loop which allows instructions to be run 0 or more times.

While loops are useful for running code over and over in situations such as while player is alive do certain tasks.

Repeat until loop

After running the body of the code once it will loop the code until the condition is met.

Repeat until loop is a Post-test loop which will run one or more times.

Example would be to use it for running a program until window close is requested.