Variables

Definitions

- **A variable** is a place in the memory dedicated to storing a value which has the ability to change, throughout the execution of a program, depending on certain conditions or information passed into the computer system.
- **A constant** is a place in the memory which is dedicated to storing a value that is fixed and assigned in the code.
- **Scope** relates to the sections of source code where the variable is defined. Variables can be global or local.
- **Identifiers:** The unique names assigned to variables & constants as a way of identifying them.

Declarations

- A variable can change through the execution of the program due to a line of code that alters its value. The assignment overwrites the previously stored value.
- Declarations are named constants/variables that also determine the data type and where it will be stored in the memory.

Identifiers

These can be made unique in two different ways:

1) Using unique names in their declaration
2) Utilising different scopes within the program. These rules determine where an identifier is defined. Local variables/constants are only defines by their identifier within the function they are created in. Conversely, the global variables/constants given an identifier are defined all through the program.

The scope of variables

- **Local variables** are variables that are created within, and will only operate in a section of code/sub-routine (e.g. a loop, procedure, class). They are created when the routine is called and destroyed when it ends.

…..Variables are declared within a function when they are only needed for use within that routine.

- **Global variables** are variables that are created outside of sub-routines, visible and so accessible to any part of the code within the program. They are created when the program starts and destroyed when it ends.

…..Variables are to be defined globally only if they are required to be accessed all through the source code and do not conflict with any identical identifiers.

Programs can be safely divided into components where the identifiers within are only specific to that division of source code. This prevents conflicts between different parts of programs/other programs being executed ; the same names for identifiers can be used in different places to refer to different things. These types of identifiers have a module scope: only being defined throughout a single module or file.