The validation of the second part is straightforward because it always has the same format (NAA).

An algorithm for validating this part of the postcode is:

```
IF the length of the second part is three THEN
    IF 1st character is a digit AND 2nd character is a letter AND the 3rd character is a letter THEN
        Second part of the postcode is valid
    ELSE
        Second part of the postcode is invalid
    END IF
ELSE
    Second part of the postcode is invalid
END IF
```

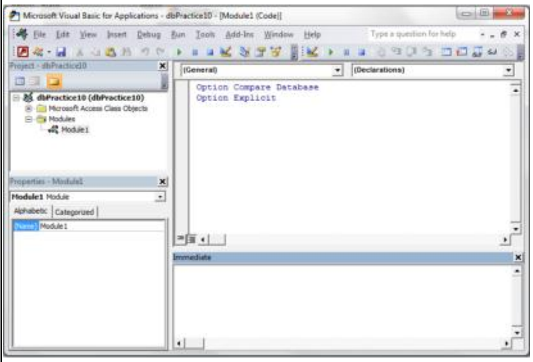
The Solution

The solution describes how to create and set up the module and Visual Basic code to validate a postcode. This should only be attempted if you have experience in using Visual Basic, since a single typing error will result in code that will not work properly.

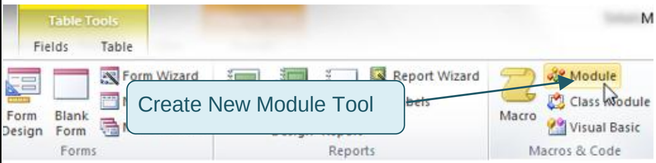
You can bypass the typing by importing the module from the sample database *dbSolution10accdb*. This method is described in Appendix A ↪.

Step-by-Step

This section explains how to create the module and enter the Visual Basic code necessary for postcode validation. The code, together with an explanation of how it works, is given at the end of the solution on page 10-8 onwards ↪. Make a copy of the database file *dbPractice10.accdb* to use as a starting point if you want to follow the solution through step-by-step.


Microsoft Visual Basic for Applications - dbPractice10 - [Module1 (Code)]
Project - dbPractice10
dbPractice10 (dbPractice10)
Microsoft Access Class Objects
Modules
Module1
Properties - Module1
Module1 Module
Alphabetic
Categorized
(General)
(Declarations)
Immediate

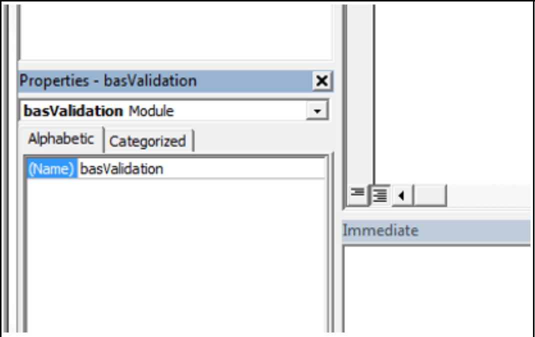
```
Option Compare Database
Option Explicit
```



Select the *Create* tab on the ribbon and click the Module tool from the *Macros & Code* group to create a new module.

By default, this will be named *Module1* The Visual Basic editor will open, displaying the code window. You may also see the project window on the left and, below it, the properties window.

If any of these windows is missing then they can be revealed using the *View* menu.



(Name) basValidation

Make sure that the properties for *Module1* are displayed by selecting *Module1* in the project window. The module has only one property: its name.

Change the name of the module in the properties window so that it is called *basValidation*. You can, of course, choose any name for the module, but the name chosen should reflect the purpose of the code that will be stored in the module.

Use *File/Save* to save the module.