**Exercise 1: Display a phrase in the log window**

*Aims:*

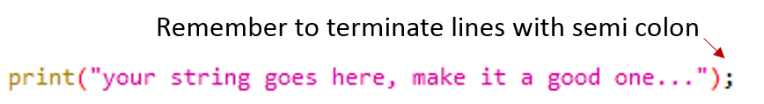
* Here we will create a macro which makes use of two important concepts in programming:

1. A function
2. A string

* The aim of this exercise is to print a simple message to the log window

*Syntax:*

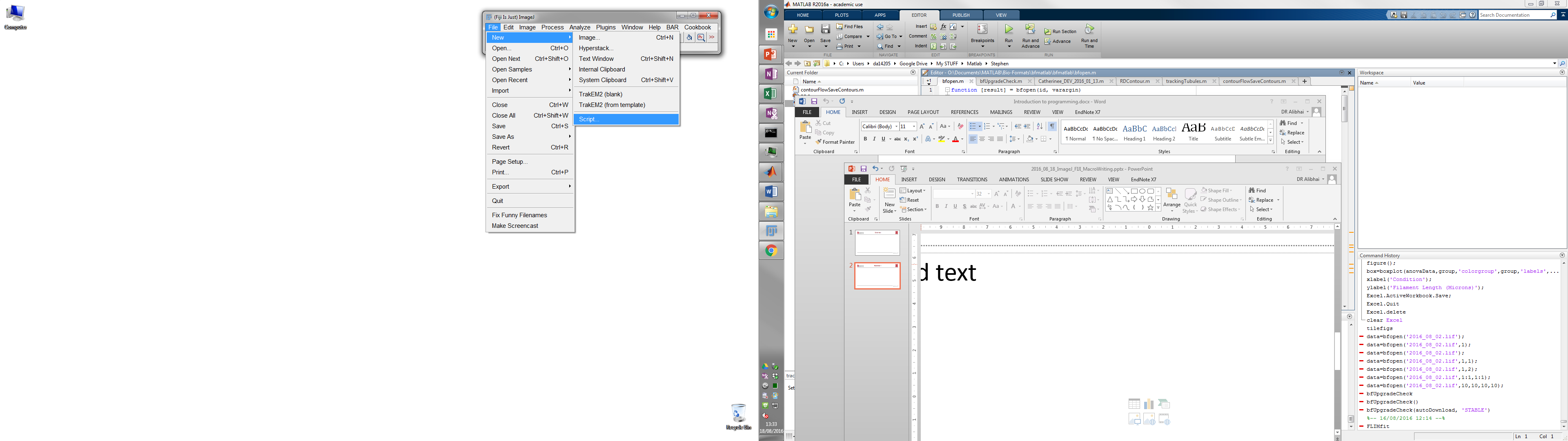
* The print function has the following syntax:



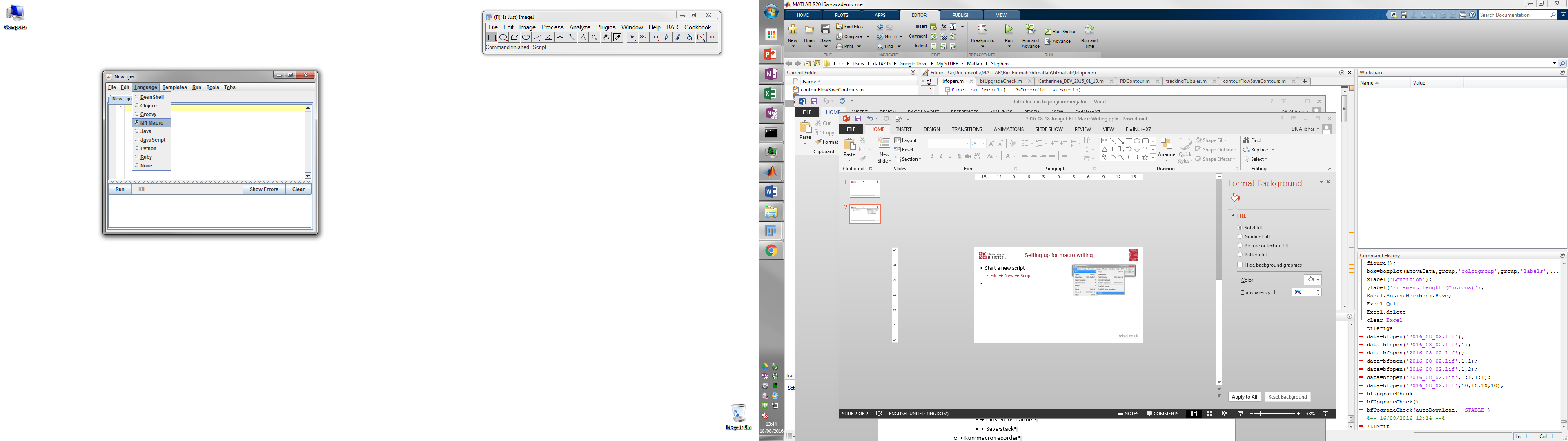
* Strings are identified using speech marks, you will notice the text turns pink once the programme recognises the words as a string (this will only happen upon closing the speech marks):
  + “This is a string”
  + This is not a string

*Helpful tips:*

* Start a new script by going to File🡪New🡪Script



* Change the programming language to ImageJ macro by going to Language🡪IJ1 Macro



* In line 1 of the script window insert a print function
* Within the brackets of the function insert a string you wish to print to the log window
* Don’t forget to terminate the function using a semi colon!
* Click Run.

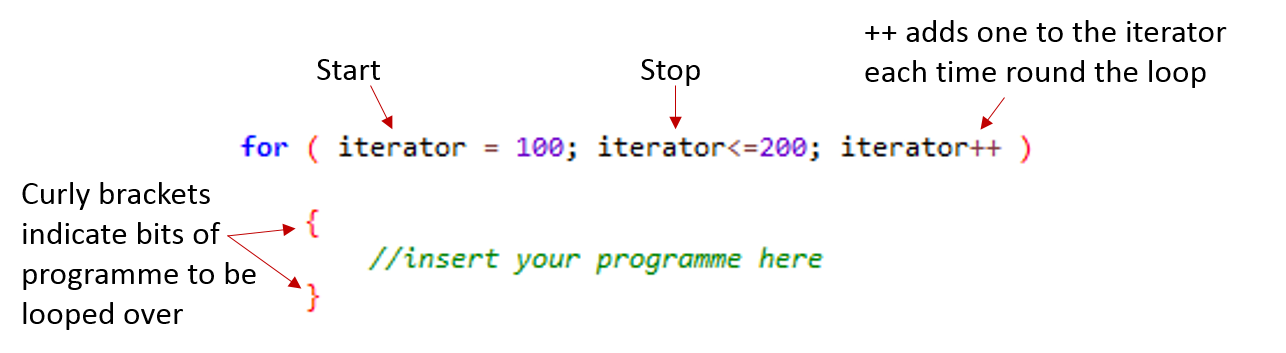
**Exercise 2: Determine if a number is odd or even**

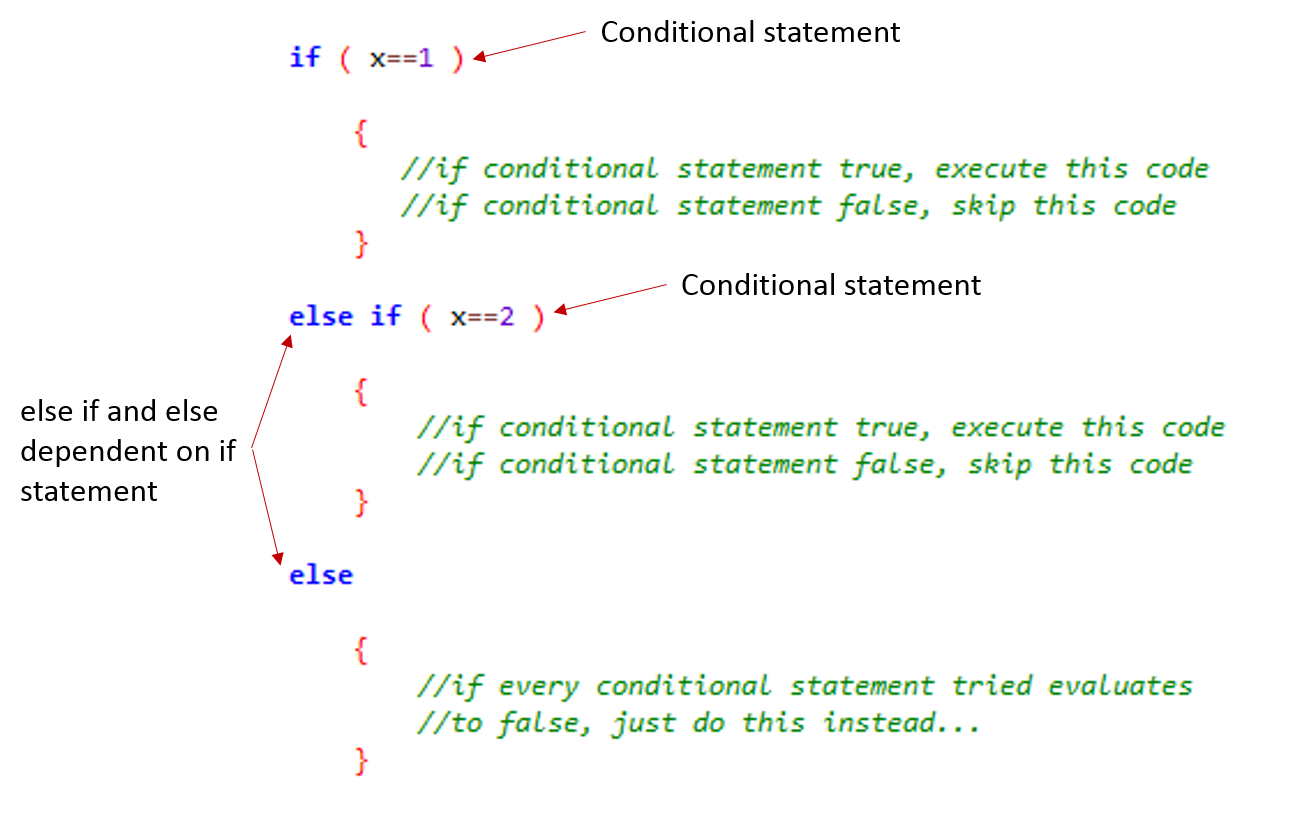
*Aims:*

* This exercise introduces the use of more complex programming concepts which will be used in almost all programmes:
  1. A loop
  2. If/Else statements
  3. Printing different statements depending on the answers to If/Else statements
* Your macro should eventually print a list of numbers and determine if they are even or odd
  + First, we’ll create a for loop to list the numbers 1 to 10 in the log window
  + Then we’ll add some conditional statements to determine if the number is odd/even
  + Finally, well print a phrase which will say if the number is odd/even

*Syntax:*

* The syntax of the functions is as follows:

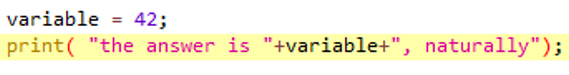




* Linking strings together can be achieved with the ‘+’ function:



* This also works for linking strings to the contents of variables:



* The operator % can be used to give the *remainder* of a division
  + 2%2 = 0 (2 divided by 2 = 1 i.e. no remainder)
  + 3%2= 1 (3 divided by 2 =1.5 i.e. not a whole division)

*Helpful tips:*

* Start a new script File🡪New
* Start by creating a loop which runs through the numbers 1 to 10:
  + Set up a for loop
  + Pick an iterator
    - This can be anything, typically we use ‘i’, but anything will do!
    - The iterator picked is then a variable assigned the start value and changes by 1 each loop.
    - Add a print function to display the iteration number
* Add in conditional statements (if/else if) to decide if the number is even or odd
  + Remember the % operator – although other methods also work…
* Add in a print statement to display a phrase indicating if the current iteration is even or odd

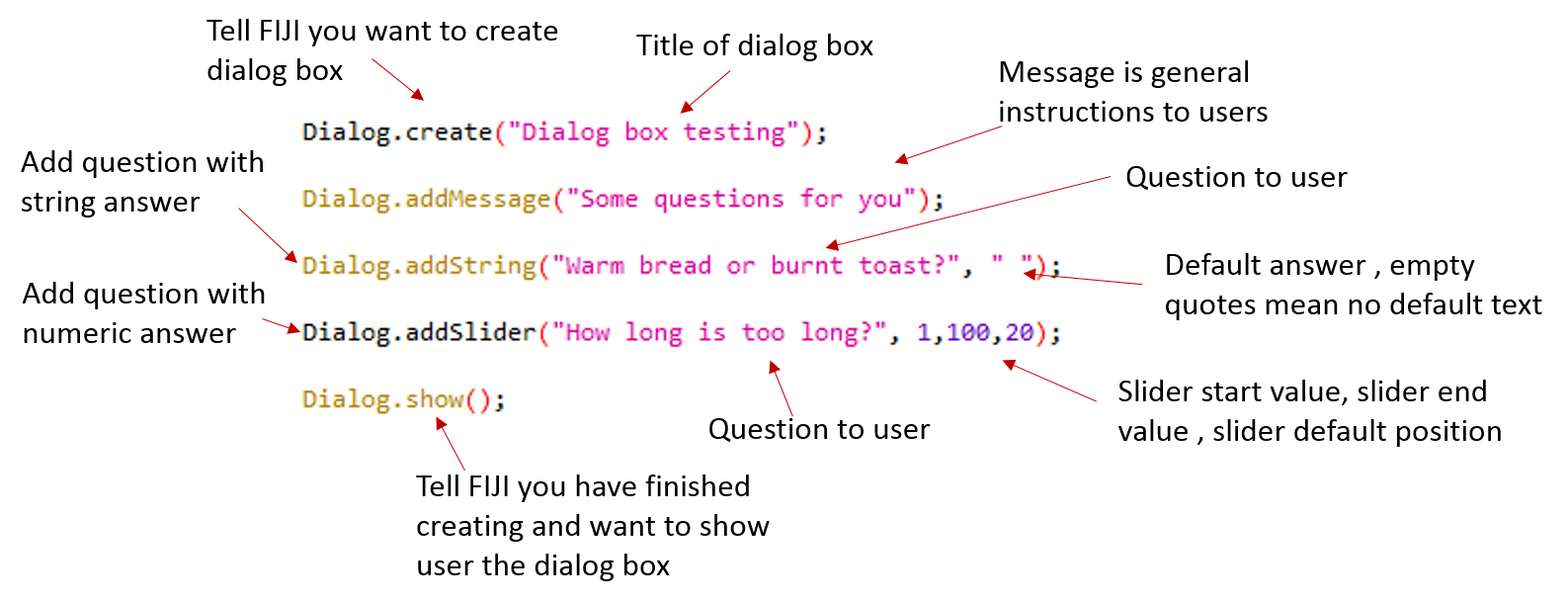
**Exercise 3: Working with simple dialog boxes**

*Aims:*

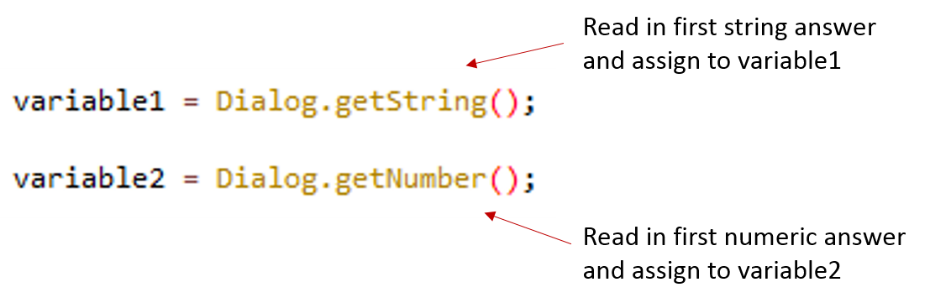
* This exercise introduces the use of dialog boxes enabling the user to input variables to the programme (e.g. threshold value, size sieve etc.).
  + The aim of this exercise is to create a dialog box which will pose several questions to the user and allow them to input data to the programme
  + Create at least two questions (one string answer and one numerical answer) For examples:
    - Question 1: “What day of the week is it?”
    - Question 2: “What date is it?”
    - Question 3: “What month is it?”
  + Read these values in and assign to variables
  + Display a concatenated answer in the log window, for the questions above a concatenated output should read:
    - Wednesday 14th September (although you can add any extra text you like too)

*Syntax:*

* The syntax needed for creating dialog boxes is as follows:

****

* Creating a dialog box using these options will allow the user to input their own values into your programme
* However, we need to add some more code to “read in” these values and assign them to variables.
* The syntax for reading in the user inputted values is as follows:

****

*Helpful tips:*

* Start a new script: File🡪New
* Create a dialog box and give it a title of your choice
* Add in a few Options for user inputs
  + A mixture of strings and numerical values would be good
* Use the Dialog.show function to finish the dialog options.
* Add the necessary lines to read in data from your dialog box
  + Remember these will have to be different depending on if the answers to the questions are numerical or string
    - The order of the read in statements will have to be the same as the order the questions were posed
  + Assign each answer to a variable
* Print the output to the log window
  + E.g. if you’re wanting to display todays date then you should be printing something like:
    - “The date today is Wednesday 14th September”

**Exercise 4: Using the Macro recorder**

*Aims:*

* With the macro recorder running, process the image using several functions (see details below)
* Run macro to process image stack automatically

*Helpful tips:*

* Manual pre-processing of image stack
  + Download image “Example Data.tif” from <http://goo.gl/urxwo9> if haven’t already done so
  + Load image
* Run Macro Recorder
  + Plugins🡪Macro🡪Record
* Process image stack:
  + Z-Project
    - Image🡪Stacks🡪 Z Project🡪Max Intensity
  + Filter
    - Process🡪Filter🡪Median
  + Threshold
    - Image🡪Adjust🡪Threshold🡪Apply
    - At this point you should see black nuclei on white background, if you see the opposite go to Edit🡪Invert
  + Separate touching objects
    - Process🡪Binary🡪Watershed
  + Analyse objects
    - Analyze🡪Analyze Particles
    - Show count masks
    - Display results
  + Change look up table
    - Image🡪look up table🡪glasbey
* Click ‘create’ in macro recorder
  + A new macro script should be created in a new window.
* Close all images apart from the original z-stack
* Click ‘run’ on macro
  + Should repeat all your manual processing…

**Exercise 5: Getting serious…adding GUI elements to your image segmentation macro.**

*Aims:*

* Take the code from exercise 4 and add GUI options
* Add option to change threshold level
* Add option to change filter type

*Syntax:*

* The basic functions for working with dialog boxes are shown below:



*Helpful tips:*

* Start by copying image analysis code from exercise 4 into a new macro script
* Identify the parts of the script that you want to make variables
  + Do they need to be of type string or numeric?
* Create a dialog box above your image analysis code
  + Insert a useful message telling the user what to do
  + Insert relevant questions to change the threshold and filter type
    - Changing filter type requires the creation of an array of pre-determined filters
  + Remember to insert the Dialog.show() statement when you’ve finished defining your dialog box
* Insert the relevant statements to read in the required information
  + One will need to be able to read in information from your array of choices
    - The necessary function to read in from an array is: Dialog.getChoice()
  + The other will need to be able to read in information in a numeric format.
  + The order these appear will need to match the order you asked the questions.
  + Assign this information to variables
  + Switch in these variable names into your image analysis code in place of the threshold value and filter type.

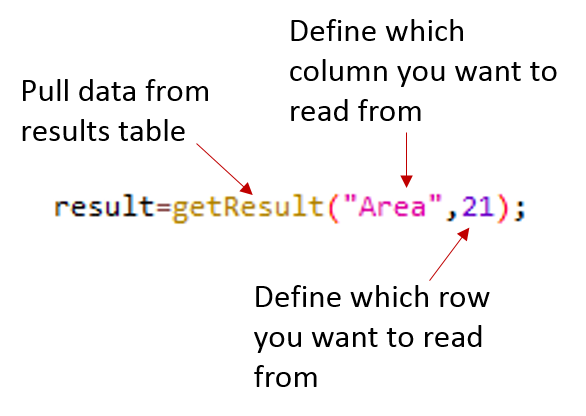
**Exercise 6: Calculating the distance between two objects**

*Aims:*

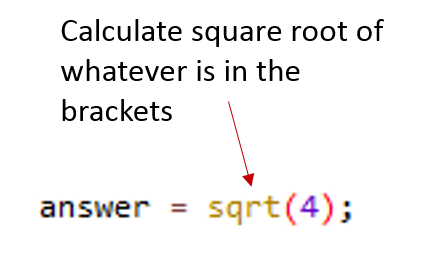
* Threshold nuclei image to create binary mask
* Use Analyze particles function to find X and Y coordinates and store in results table
* Read in coordinate information from results table and calculate straight line distance between the two nuclei

*Syntax:*

* To get a result stored in the results table and store it in a variable:



* To calculate the square root of a number or variable:



*Helpful tips:*

* Load in image of two nuclei (“two\_nuc.tif” from <http://goo.gl/q0OUp5>)
* Apply gaussian blur filter
* Apply manual threshold and adjust until both nuclei selected.
* Convert to binary mask.
* Go to Analyze🡪set measurements and ensure that “Centroid” is selected
* Use Analyze 🡪 Analyze Particles to measure nuclei size and centroid position
  + This information should now be stored in the results table
* Open a new scripting window
* Get the x and y coordinates of nuclei 1 (remembering macros use zero based indexing…)
* Get the x and y coordinates of nuclei 2
* Calculate the distance between these two points using:
* Print the calculated distance to the log window.

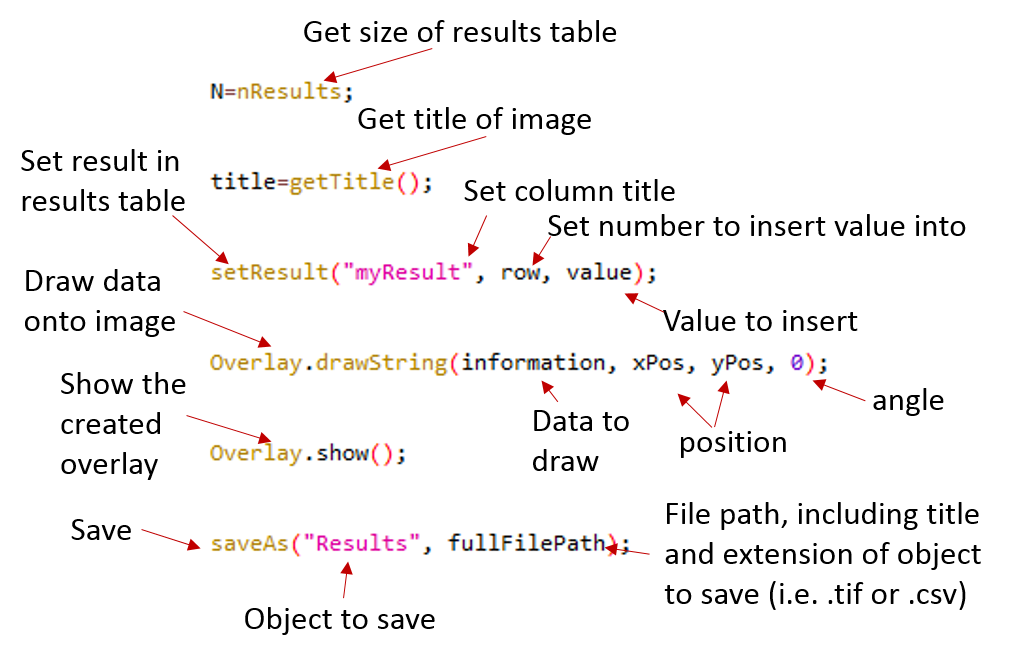
**Exercise 7: Putting everything together**

*Aims:*

* Create a macro to segment and measure the nearest neighbour distance for each nucleus in “Example\_Data.tif”
* Include a GUI for the user to input options (threshold, filter type, filter size etc.)
* Update the results table to show the calculated nearest neighbour distance in the results table and save table as .csv file.
* Display the nearest neighbour distance on the segmented image

*Syntax:*

* Some useful functions for this exercise can be found below:



*Helpful tips:*

* A good start is to take the result of exercise 5 as a starting point
* Extend the logic from exercise 6 using nested for loops (i.e. 2 “for” loops, one inside the other) to scan through each nucleus and calculate the distance to every other nucleus, keeping the value of the nearest.
* Add the nearest neighbour distance to the results table
* Save the results table to a .csv format which can then be opened in Excel.