

# **FACULTY OF ENGINEERING AND THE BUILT ENVIRONMENT**

**DEPARTMENT OF ELECTRONIC ENGINEERING**

**Practical Manual**

**BETxxx Computing and IT**

**2020**

**SESSION 1: Computer (PC) hardware investigation**

**Session outcome**

In this practical, learners will be given an opportunity to remove all the components of a desktop computer (PC), locate the manufacturer’s specifications in order to identify the important components of the PC and investigate/research their important role in the overall functioning of the PC.

**Practical Assessment**

This practical assessment contributes 10% of the overall course.

Assessment criteria for this practical task are given below. Although practical work will take place in groups, each student will be assessed individually.

**Tasks**

* Ensure the PC is NOT connected to the mains power
* Carefully remove the screws holding the tower case of the PC together
* Unhinge the cover and expose the inside of the PC
* Using the serial number stamped on the mother-board, and making use of the internet, find a software copy of the manufacturer’s specification
* Based on the specifications, identify the main components of the PC
* Briefly explain to the lecturer how these components interact in a fully functional PC
* Close up the tower of the computer as was initially found

**First Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Surname: \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Group: \_\_\_\_\_**

**Student number: \_\_\_\_\_\_\_\_\_\_\_\_**

**Session 1 Assessment Rubric:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Assessment Component** | **Weight (%)** | | | | | |
| **Totally correct** | **Mostly correct** | **Partially correct** | **Mostly incorrect** | **Not attempted** |  |
| Identify the main components of the PC | 20 | 15 | 10 | 5 | 0 |  |
| Provide specifications of the RAM and microprocessor on the motherboard | 20 | 15 | 10 | 5 | 0 |  |
| Observe operation and answer questions posed | 60 | 45 | 30 | 15 | 0 |  |

**SESSION 2: Introduction to report writing, virus and anti-plagiarism checking**

**Session outcome**

At the end of this session, the student would have learnt:

1. Performing relevant tasks using a word processor, related to the compilation of an engineering report
2. Working with the language tool to check for spelling errors
3. Scanning a document/folder to ensure it is free of virus
4. Performing an originality check using Turnitin software

**Tasks**

* Open a blank document in Microsoft Word
* Type a short paragraph and format it using the guidelines stipulated in the department’s report writing guideline (available on Blackboard)
* Obtain a word count for your document and undertake a spell check of the document
* Save the document on your USB stick
* Run an anti-virus check on the usb to ensure it is free of any virus
* Perform an originality check on your document using Turnitin
* Make relevant changes to your document to ensure the overall plagiarism total is less than 3% from any of the sources highlighted in the Turn-it-in report
* Present the results thereof, to the lecturer
* Log off Turnitin

**SESSION 3 & 4: Basic formulae design in Excel**

**Session outcome**

At the end of this session, the student would learn:

1. Design of basic mathematical formulae in Excel
2. Application of conditional statements in solving engineering problems
3. Plotting of basic graphs using Excel

**Tasks**

* Open a blank document in Microsoft Excel
* Enter the table below into your document and save it



* Determine the appropriate formulae for the row TOTAL A, columns TOTAL B, Cheapest, Most expensive and Average Price
* Modify the table as shown below



* Now write down the formulae in the Description columns for each of the 3 suppliers so that depending on the price of the component, the words either “cheap” or “expensive” are displayed in the applicable cell location.
* Open up a blank document in Excel and copy the following 2 tables



* Now plot an XY graph to obtain the graph as shown below:

**SESSION 5 & 6: Financial analyses incorporating amortisation**

**Session outcome**

At the end of this session, the student would learn:

1. the importance of amortisation calculations in engineering.
2. the use of the PMT function in Excel to undertake amortisation calculations
3. Using graphs to develop a better understanding of amortisation

**Practical Assessment**

This practical assessment contributes 10% of the overall course.

Assessment criteria for this practical task are given below. Each student will be assessed individually.

**Problem Statement**

As the team leader of an engineering project, you are responsible for conducting the financial analysis pertaining to the purchase of a specialised laptop computer required to successfully complete the project.

The cost of the laptop is R24000. This amount needs to be paid over 5 years (in monthly instalments made at the end of each month), at a fixed annual interest rate of 10%.

**Tasks**

1. Design an Excel spreadsheet with the loan data entered as follows:

|  |  |
| --- | --- |
| **Loan Data** | |
| Loan Amount | **R 24 000.00** |
| Annual Interest Rate | **10.00%** |
| Loan Period in Years | **5** |
| Number of Payments Per Year | **12** |
| Start Date | **28-Feb-2018** |

1. Enter the following cells into the spreadsheet:

|  |  |  |
| --- | --- | --- |
| **Summary** | | |
|  | Payment (per period) | **509.93** |
|  | Number of Payments | **60** |
|  | Actual Number of Payments | **60** |
|  | Total Interest Paid | **6 595.77** |
|  | Total % Interest | **0.27** |
|  | Total Extra Payments | **R 0** |
|  | Total Payment | **30 595.77** |

1. Design the appropriate PMT formula in the cell labelled “Payment (per period)”.
2. Complete the following table in the spreadsheet for each of the 60 months

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Payment No.** | **Payment Date** | **Payment amount** | **Principal amount** | **Interest amount** | **Extra Payments** | **Balance** |
|  | **28-Feb-2018** |  |  |  |  | R 24 000.00 |
| 1 | 28-Mar-2018 | **509.93** | **309.93** | **200.00** | **0** | **23 690.07** |

1. Go back to the table in step 2 and enter the formula to determine the contents of the cells labelled “total interest paid”, “total % interest” and “total payment”
2. Draw a graph of the principal amount paid versus the interest portion paid. The graph should look similar to the one below:
3. Determine from the spreadsheet analysis, at which month the principal and interest amounts paid become equal.

**First Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Surname: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Group: \_\_\_\_\_**

**Student number: \_\_\_\_\_\_\_\_\_\_\_\_**

**Session 6 Assessment Rubric:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Assessment Component** | **Weight (%)** | | | | | |
| **Totally correct** | **Mostly correct** | **Partially correct** | **Mostly incorrect** | **Not attempted** |  |
| Implementation of Excel spreadsheet for amortisation calculation | 20 | 15 | 10 | 5 | 0 |  |
| Design and use of Excel formulae | 20 | 15 | 10 | 5 | 0 |  |
| Determination & identification of break-even point | 60 | 45 | 30 | 15 | 0 |  |

**SESSION 7: Introduction to Matlab programming**

**Session outcome**

At the end of this session, the student would learn:

* To use the Matlab environment
* To compile a flowchart and write and test a simple program
* To appropriately modify a program to obtain the desired output and functionality
* The basic use of mathematical functions in Matlab programming

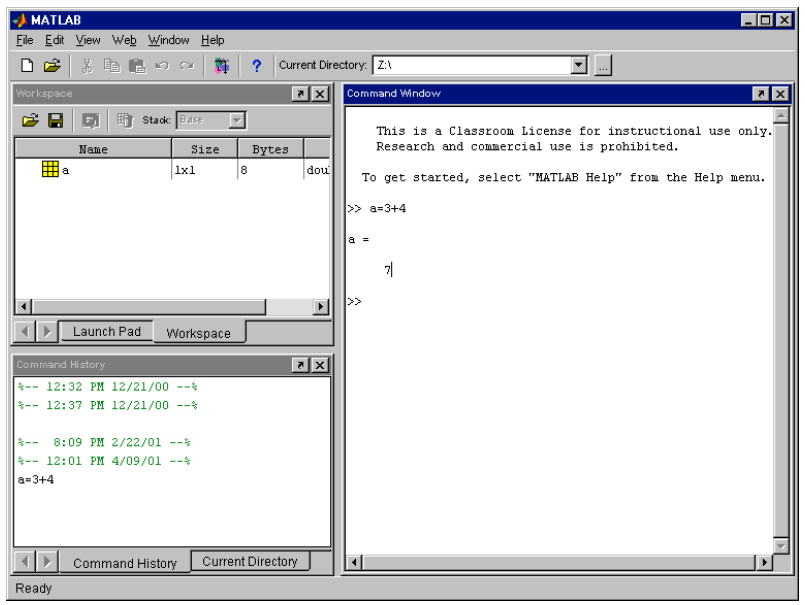
Introduction: Using MATLAB for general maths

MATLAB, short for MATrix LABoratory is a programming package specifically designed for quick and easy scientific calculations and I/O. It has literally hundreds of built-in functions for a wide variety of computations and many toolboxes designed for specific research disciplines, including statistics, optimization, solution of partial differential equations, data analysis.

**• Section 1 – The Basics**

1) Start MATLAB by double clicking on the MATLAB icon

The MATLAB window should come up on your screen. It looks like:



This is the window in which you interact with MATLAB. The main window on the right is called the Command Window. You can see the command prompt in this window, which looks like >>. If this prompt is visible MATLAB is ready for you to enter a command. In the figure, you can see that we typed in the command a=3+4.

In the top left corner you can view the Launch Pad window and the Workspace window. Swap from one to the other by clicking on the appropriate tag. The Workspace window will show you all variables that you are using in your current MATLAB session. In this example, the workspace contains the variable 'a'. When you first start up MATLAB, the workspace is empty.

In the bottom left corner you can see the Command History window, which simply gives a chronological list of all MATLAB commands that you used, and the Current Directory window which shows you the contents and location of the directory you are currently working in.

**• Section 2 - The MATLAB on-line help facility**

MATLAB has handy on-line help facilities. There are several ways to get help. You can go to Help on the menu (or the ? on the menu) and select any of the available help facilities listed there. There are simpler help commands as well that work in all versions of MATLAB.

1) Type the command help in the Command Window to find a long list of all different categories for which there are MATLAB commands. Each of the listed categories contains more detailed information about available MATLAB functions.

**EXERCISES FOR PRACTICAL**

1. Perform the operation 3\*4+6.
2. Perform the addition operation 7+9.
3. Perform the subtraction operation 16-10.
4. Perform the multiplication operation 2\*9.
5. Perform the division operation 12/3.
6. Perform the division operation 12/5.
7. Perform the exponentiation operation 3^5.
8. Perform the exponentiation operation 3\*(-5).
9. Perform the exponentiation operation (-3)^5.
10. Perform the exponentiation operation -3^5.
11. Perform the multiple operations 5+7-15.
12. Perform the multiple operations (6\*7)+4.
13. Perform the multiple operations 6\*(7+4).
14. Perform the multiple operations 4.5 + (15/2).
15. Perform the multiple operations (4.5 + 15)/2.
16. Perform the multiple operations (15 – 4 + 12)/5 – 2\*(7^4)/100.
17. Perform the multiple operations (15 – 4) + 12/5 – (2\*7)^4/100.
18. Perform the fraction addition (2/3) + (3/4) numerically.
19. Obtain help on the cos command.
20. Perform the operation cos(5) . The value of 5 is in radians.
21. Obtain help on the pi command.
22. Compute the value of . *(Note that pi is already defined in Matlab)*
23. Assign the value of 5.2 to the variable y.
24. Obtain help on the sqrt command.
25. Perform the operation 3\* 6 +*x* for *x* = 4 .
26. Assign the values of 3 and 4 to the variables x and y, respectively, then calculate the value of z where *z* = 2*x* −7*y* .

**SESSION 8: Introduction to data types, reading inputs, number systems and Boolean functions**

**Session outcome**

At the end of this session, the student would learn:

* To compile a flowchart and write a basic program
* To appropriately modify a program to obtain the desired output and functionality
* The importance of data types, reading inputs, number systems and Boolean functions in Matlab programs

Tasks

1. Perform the operation 2\*3+7 and store the result in the variable w.
2. Define the three variables a, b, and c equal to 4, -10, and 3.2, respectively.
3. Define the two variables y and Y equal to 10 and 100. Are the two variables identical?
4. Let x = 5.5 and y = -2.6. Calculate the value of the variable z = 2x-3y.
5. In Exercise 4 above, calculate the value of the variable w = 3y – z + x/y.
6. Let r = 6.3 and s = 5.8. Calculate the value of the variable final defined by final = r + s - r\*s.
7. In Exercise 6 above, calculate the value of the variable result defined by result = r^2 – s^2.
8. Define the three variable width, Width, and WIDTH equal to 1.5, 2.0, and 4.5, respectively. Are these three variables identical?
9. Write the following comment in MATLAB: This line will not be executed.
10. Assign the value of 3.5 to the variable s then add a comment about this assignment on the same line.
11. Define the values of the variables Y1 and Y2 equal to 7 and 9 then perform the calculation Y3 = (Y1 – Y2)/3.
12. Perform the operation 2\*m – 5. Do you get an error? Why?
13. Define the variables cost and profit equal to 175 and 25, respectively, then calculate the variable sale\_price defined by sale\_price = cost + profit.
14. Define the variable centigrade equal to 28 then calculate the variable fahrenheit defined by fahrenheit = (centigrade\*9/5) + 32.
15. Perform the who command to get a list of the variables stored in this session.
16. Perform the whos command to get a list of the variables stored in this session along with their details.
17. Clear all the variables stored in this session by using the clear command.
18. Calculate both the area and perimeter of a rectangle of sides 5 and 7. No units are used in this exercise.
19. Calculate both the area and perimeter of a circle of radius 6.45. No units are used in this exercise.
20. Compute the square root of 10.
21. Compute the factorial of 7.
22. Compute the cosine of the angle 45 where 45 is in radians.
23. Compute the cosine of the angle 45 where 45 is in degrees.
24. Compute the sine of the angle of 45 where 45 is in degrees.
25. Compute the tangent of the angle 45 where 45 is in degrees.
26. Compute the value of exponential function *e*3 .
27. Compute the value of the natural logarithm ln 3.5 .
28. Compute the value of the logarithm log10 3.5.
29. Use the MATLAB rounding function round to round the value of 2.43.
30. Use the MATLAB remainder function rem to obtain the remainder when dividing 5 by 4.
31. Compute the absolute value of -3.6.
32. Compute the value of the expression 1.5−2 6.7/5
33. Compute the value of sin2 pi+cos2 pi.
34. Compute the value of log10 0 . Do you get an error? Why?

**Session 9 and 10: Data representation in Matlab**

**Session outcome**

At the end of this session, the student would learn:

* To compile a flowchart and write a basic program
* To appropriately modify a program to obtain the desired output and functionality
* The importance of data representation within programs

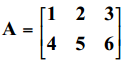
**Data representation in MATLAB**

MATLAB stands for 'MATrix LABoratory'. This title is appropriate because the structure for the storage of all data in MATLAB is a matrix. The MATLAB matrix-variables may have any number of rows and columns. Scalars like the variables a and b that you worked with above are also stored as matrix variables with 1 row and 1 column. This is the reason that the variable a in your workspace is shown as a 1x1 matrix!

So beware, a matrix-variable can be any variable in MATLAB, that is, it could be a scalar, a vector or a matrix of any size. If we refer to scalars, vectors or matrices specifically we mean just that: scalars, vectors or matrices.

**Entering variables**

An mxn ('m by n') MATLAB matrix-variable (or simply variable) has m rows and n columns. For example, the variable

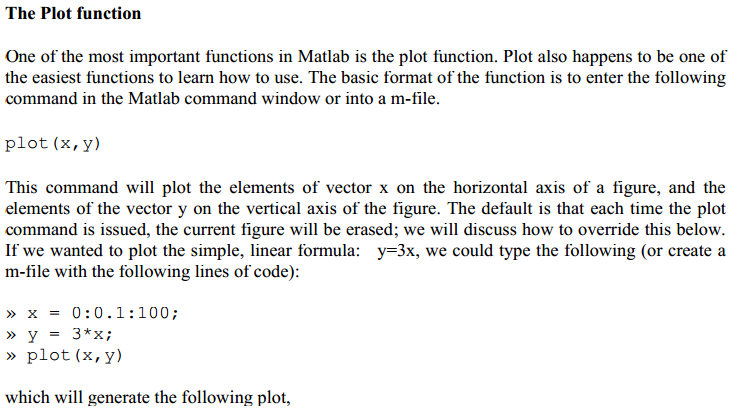


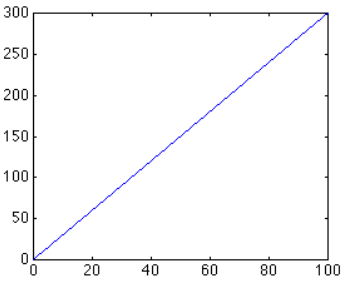
A is a 2x3 matrix. The numbers 1 through 6 are called the elements of the matrix. The element on row 1 and column 2 has the value 2. The element on row 2 and column 3 has the value 6.

1) Create the matrix A by typing A = [1 2 3; 4 5 6] followed by enter. Make sure that you separate the elements 1,2 and 3, and 4, 5 and 6 with a space. If you don’t MATLAB will think you created the vector A with first element 123 (one hundred and twenty-three) and second element 456. Also, use square brackets instead of parentheses.

Instead of a space, you can also use a comma to separate the elements. The semi-colon (;) in this context is used to separate the elements of one row from another, but you can also use a line break to separate rows as shown below.

**Plotting a Graph**





Exercises

1. Generate the following matrix: A



|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |

1. Generate the following vector b



1. Evaluate the vector c where {*c*} =[*A*]{*b*} where A is the matrix given in Exercise 1 above and b is the vector given in Exercise 2 above.
2. Solve the following system of simultaneous algebraic equation using Gaussian elimination.



1. Solve the system of simultaneous algebraic equations of Exercise 4 above using matrix inversion.
2. Generate the following matrix X:



1. Extract the sub-matrix in rows 2 to 3 and columns 1 to 2 of the matrix X in Exercise 6 above.
2. Extract the second column of the matrix X in Exercise 6 above.
3. Extract the first row of the matrix X in Exercise 6 above.
4. Extract the element in row 1 and column 3 of the matrix X in Exercise 6 above.
5. Generate the row vector x with integer values ranging from 1 to 9.
6. Plot the graph of the function *y*=*x*3 −2 for the range of the values of x in Exercise 11 above.

**Session 11: Running and debugging Matlab programs**

**Session outcome**

At the end of this session, the student would learn:

* To compile a flowchart and write a basic program
* To appropriately modify a program to obtain the desired output and functionality
* To run and debug Matlab programs

**Running and debugging a program**

To help you see what a script looks like and how to run and debug it, here is a simple one that asks for a small step-size h along the x-axis, then plots the function f(x) = e−x/6 \*cos x from x = 0 to x = 20. The script then prints the step-size h and tells you that it is finished.

Type this script into an M-file called test.m using Matlab’s editor with the desktop arranged as shown on the previous page. Save it, then run it by typing test

in the command window. Or, alternatively, press F5 while the editing window is active and the script will be saved, then executed. Run the sample script below three times using these values of h: 1, 0.1, 0.01. As you run it look at the values of the variables h, x, and f in the workspace window at the upper right of the desktop, and also examine a few values in the array editor just below it so that you understand what the array editor does.

**Sample script:**

clear; % clear all variables from memory

close all; % close all figure windows

h=input(’ Enter the step-size h - ’) ;

x=0:h:20; % build an array of points [0,h,2h,...,20]

f=exp(-x/6).\*cos(x); % build the array [f(0),f(h),...f(20)]

plot(x,f)

fprintf(‘Plot completed, h = %g \n’, h)

**1.Breakpoints and Stepping**

When a script doesn’t work properly, you need to find out why and then fix it. It is very helpful in this debugging process to watch what the script does as it runs, and to help you do this, Matlab comes with two important features: breakpoints and stepping.

To see what a breakpoint does, put the cursor on the x=0:h:20 line in the sample script above and either click on Breakpoints on the tool bar and select Set/Clear, or press F12.

Now press F12 repeatedly and note that the little red dot at the beginning of the line toggles on and off, meaning that F12 is just an on-off switch to set a breakpoint. When the red dot is there it means that a breakpoint has been set, which means that when the script runs it will execute the instructions in the script until it reaches the breakpoint, and then it will stop. Make this happen by pressing F5 and watching the green arrow appear on the line with the breakpoint. Look at the workspace window and note that h has been given a value, but that x has not. This is because the breakpoint stops execution just before the line on which it is set.

Now you can click on the Debug icon on the tool bar to see what to do next, but the most common things to do are to either step through the code executing each line in turn (F10) while watching what happens to your variables in the workspace and array editor windows, or to just continue after the breakpoint to the end (F5.) Take a minute now and use F10 to step through the script while watching what happens in the other windows.

When you write a script to solve some new problem, you should always step through it this way so that you are sure that it is doing what you designed it to do. You will have lots of chances to practice debugging this way as you work through more examples.

**Session 12: Matlab programming**

**Session outcome**

At the end of this session, the student would learn:

* To compile a flowchart and write a basic program

**Exercises**

1. Write a script of four lines as follows: the first line should be a comment line, the second and third lines should have the assignments cost = 200 and sale\_price = 250, respectively. The fourth line should have the calculation profit = sale\_price – cost. Store the script in a script file called example.m.

Now, run the above script as follows:

>> example

Verify that the answer is correct.

1. Now modify the program above so that the user is asked to enter the values for the cost and sale\_price. Hint: use the **input** function in Matlab
2. Write a function of three lines to calculate the volume of a sphere of radius r. The first line should include the name of the function which is volume(r). The second line should be a comment line. The third line should include the calculation of the volume of the sphere which is ∏*r*3 . Store the function in a function file called volume.m then run the function with the value of r equal to 2

Now, run the above function as follows:

>> volume(2)

Verify the answer is correct.

1. Now modify the program above so that the user is asked to enter the value for the radius r. Hint: use the **input** function in Matlab. Also use the disp and fprintf functions.
2. Write a function with two arguments to calculate the area of a rectangle with sides a and b. The function should have three lines. The first line should include the name of the function which is RectangleArea(a,b). The second line should be a comment line. The third line should include the calculation of the area of the rectangle with is the product a\*b. Store the function in a function file called RectangleArea.m then run the function twice as follow: the first execution with the values 3 and 6, while the second execution with the values 2.5 and 5.5.

**Session 13: Additional programs**

**Session outcome**

At the end of this session, the student would learn:

* To compile a flowchart and write a basic program
* To understand and design programs using the for, do and while statements

**Programs**

1. a=80 and b=40. Write a program to display the answers for the sum, difference, multiplication, division and modulus of the two numbers. Hint: use the fprintf and disp functions as required.
2. Modify the program above so that the user is required to enter the values for a and b. Display the results for the sum, difference, multiplication, division and modulus.
3. Request a user to enter a number. Write a program to determine if the number is even or odd. Display the result accordingly.
4. Request a user to enter a number. Write a program to determine if the number is a prime number. Display the result accordingly.