Introduction

This assignment is in two parts. Part 1 is an exercise where you need to write either one or two script programs (one in MS-DOS batch language and/or one in the Linux bash scripting language). Both programs will do the same task but in different scripting languages. This will form up 60% of the marks for this assignment (see below).

Part 2 is a set of research questions on operating system you need to look up in text books or on line and write up your answers. This second part will form the other 40% of the marks. *You need to attempt at least one of the programs in part 1 AND the research questions in part 2. You will find it hard to pass if you do not at least attempt both parts.*

Part 1: Scripting Task (worth up to 60%)

For this part you have a choice, you can choose to produce a solution in only one scripting language (either MS-DOS batch or Linux bash Shellscript) for up to 40% or in both MS-DOS and Linux bash Shellscript for up to 60%.

The task (in whichever scripting language or both) is the same, to create a “safe” editor tool which backs up the file it edits before editing it. If you decide to submit both scripting languages the programs should both run in the same way.

Whichever scripting language you choose (or both) the program(s) you produce should run in two modes, interactive and command line.

In the interactive mode your program will prompt the user for a filename and if the file entered as the filename exists in the current directory it will make a backup copy with an extension of “.bak” (overwriting any file of that name if it exists) in the current directory then go on to edit it (either with notepad if in MS-DOS or vi in Linux). If the file does not exist already it will not create a backup.

So, for example when the program runs interactively it should (after a suitable introduction) prompt the user:

What File Do You Wish to Edit?

then read in the filename from the user. If the file exists it backs it up (outputting a suitable message) then edits the file.

For the command line mode the program should take exactly one parameter on the command line rather than prompt the user for the filename. If the user gives more than one parameter on the command line the program should output an error message and exit. Other than that it should behave as the interactive version does and backup the file if it already exists with a suitable message. So, for example, if he script program was called (say, in MS-DOS) safeedit.bat then the user might type

safeedit.bat myfile.txt

and if the file myfile.txt exists a copy will be made “myfile.bak” but if the user put

safeedit.bat myfile.txt myfile2.txt

it would say something like “too many parameters entered”. For maximum marks each or both solutions must run in interactive and command line modes.

Hand in

Note – for full marks you will be expected to hand in ALL the things listed below for each version of the task you elect to produce. Failure to do so will result in lower marks even if your program(s) fully meet the above functional specifications. Marks will also be deducted for poor presentation and poor programming practice. Just handing in a disk with no printed component or a printed component and no disk will result in a low (possibly fail) mark.

When you hand in your work you should hand in the work for this task AND your answers to the “Research Question” sheets described above all bound in one folder with one e-vision cover sheet. ***Do no hand them in separately****.*

By the deadline above, you must hand in a written report containing:

- Fully commented single spaced listings of one or both programs (depending on if you decide to hand in a version in one scripting language or both). Minimally or uncommented and/or poorly laid out code is likely to score very low marks. Take care to check that Word does not mess up your code formatting (for example wrapping round code lines).

- Screen shots showing your programs running. Your screen shots should be annotated (indicating clearly what function the screen shot shows) and show that the programs work exactly as per the spec above.

- Think carefully what screen shots you show, they should be selected to show the various aspects and routes through the program all work correctly, random screenshots with little or no evidence of thought about the above or excessive screen shots where most or all do not aid proving the code works correctly are also likely to score poorly. *Remember the user is always right ☺…. It is no good implementing something other than what is asked for (including extra unnecessary features) and they trying to argue your way is better!* ***What the customer wants, the customer gets even if you disagree with them...***

- A bibliography list if you used any sources to help you develop your code cross referenced where needed to your report (including references in the research questions). If you are unclear how to reference properly there is a guide to how on the ARU Library website. It is not essential you use a specific style but you should use a recognised referencing style such as Harvard. If you directly used code from another source then this should also be stated in the comments in your programs.

- A CD/DVD or memory stick containing the (suitably named) program files (as text NOT Word files).

Tips:

- **Keep an electronic backup of all submitted work in case your script or disk goes missing!**

- *Check your CD/DVD can be read before you hand it in (maybe try to open it on another computer to the one you created it on?). If we receive a blank disk or one we cannot read we will be unable to test your code and will give you zero for that aspect which will significantly reduce the marks you can get for the assignment.*

Please make sure what you hand in is clearly your own work. If you use code or material from other sources (including the lecture notes) this should be made clear in the code comments or in your research question answers. Submissions suspected of plagiarism will be passed to the Director of Studies for investigation.

Mark Breakdown (out of the 60% for this section):

Aspect	**Maximum marks if you submit a solution in just ONE scripting language (MS-DOS or bash)**	**Maximum marks if you submit solutions in BOTH scripting languages (MS-DOS and bash)**
Well designed and written code which complies with the specification and works as required (including good code layout, error checking and commenting).	25%	40%
Appropriately selected screen shots which are legible and fully demonstrate the operation of the code.	10%	15%
Presentation of the report and referencing	5%	5%
Total Possible Marks (out of 60%)	**40%**	**60%**

Part 2: Research Questions (worth 40%)

Instructions/Information - Please Read This Carefully: You should hand in your answers to these questions in the same report as the code and screenshots etc. described above. **Do not hand in answers to these questions separately.**

- Please only type your answers, do not submit hand written answers. You should include the question text and question number in the report with the relevant answer. Copy the questions exactly, do not paraphrase or alter the text.
- All questions are worth the same (4 marks, so 10 questions make a total of 40%). You are only expected to write fairly short answers per question, perhaps just a paragraph or two.
- You should attempt all questions. Make sure you answer the question asked, only the question and nothing but the question. Writing about something other than what the question asks, however correct, wastes your time and will get no marks.
- All answers should be your own work, if you use information from other sources please reference it using an appropriate referencing style (see part 1 above for more details on how to reference and where to get more information on referencing).

Questions:

1. In the MS-DOS command prompt what does the command “driverquery /si” do? Why might it be useful if I have a problem with a system device?
2. My email system gives me a choice of pop3 or imap. I am currently using pop3. Why might I want to swap to imap?
3. What MS-DOS command prompt command should I used to find out my computer’s IP number? What is an IP number and why is it important?
4. The command in question 3 also tells me my default gateway. What is a default gateway?
5. If I believe my windows system files are corrupted or missing I can use SFC to repair them. What is SFC and how would I use it?
6. On an Apple Mac what does the “caffeinate” command do?
7. Under Linux what simple thing can I do to make a file not appear in a standard *ls* directory listing?
8. In an operating system what does it mean when two processes are said to be “deadlocked”?
9. In the bash shell what does the tilde (“~”) symbol mean when used in a command?
10. What is GNU nano? In which operating system am I most likely to use it?