COM1032 Operating Systems
Final Coursework FAQ

Introduction:

This FAQ Document does not precede the coursework document and the rubric it contains, which is what will the assessment confirm with. This document is constantly changing as I find a question from a student can serve other students, or a new idea or suggestions that can provide starting points or solutions to some obstacles. Being constantly changing does not mean the requirements are changing. This FAQ document serves the students as a set of hints, guidelines, recommendations probably or suggestions. Our brains are different from yours, and students' brains are different from each other. We cannot limit your creativity by giving you exact instructions to follow and replicate our brains in you. Those who invented the internet and the Linux OS in undergraduate projects did not confirm with all guidelines. A good assessor will also see the merit in good work and accept that to encourage the good creative work.

I think this coursework is pretty elastic. It allow less than average students (and good students who are affected by the current events and cannot focus as before) still score well by following the minimal requirements. It also allows brilliant students to go beyond expectations and be creative as they wish by giving them enough guidelines (without limitations) to enrich their educational experience. This is why the FAQ contains questions and answers across the range of the spectrum. You can feel this question is for you and another is not for you to consider.

To avoid being overwhelmed by the updates to this FAQ, you can limit yourself to reading only new questions added - the notification should contains a summary of changes that I add. You can also choose to ignore this document at all, and follow the coursework document and your understanding of its contents based on the lectures and labs and the online resources and your own reflections on them. You can only consider the answers you receive to your personal questions only without reading the answers asked by other students that might be irrelevant to your own plans and choices.
I hope you enjoy this educational experience in these unprecedented times. There are lots of support and easy options that you can benefit from. There are also lots of educational opportunities that can distract you from any potential panic the current events might be causing you.

1) **What do we submit?**
2)

You need to submit a zip file containing your source code of the OS Simulator, any data files in an input folder of your simulations, and any output files if any in an output folder.
You will also need a report containing the sections requested in the coursework document. Your report is ***expected*** to be from 5 pages, to a maximum of 20 pages, of font size 12. A brilliant student may amaze us by covering all requirements in less than 5 pages and still be awarded the marks they addressed. Another brilliant student can amaze us by a much longer than 20 pages that is all very interesting to read without redundancy and unnecessary copy and paste of definitions.
You need to focus on how you build your simulations, in line with the examples you did in the labs, and found in the online resources. Your choices of algorithms and strategies, need to be justified with basic definitions of these methods and strategies.

2) Does PSS.jar using Lab 3 simulator count towards Task 2, a) requirements?

Yes, and it provides good example of simulating real processing. Simulations are not the real thing. A process is again a line in the input file, with columns describing pid, time of arrival to ready queue, CPU Burst, priority. More explanation to inspire your simulations follows:
You can build an OS simulator that creates the input file for PSS, and call the scheduling methods for the dynamically created processes by the user. This OSSim gives a GUI (you can do a simpler command line) for the user to define processes:
https://sourceforge.net/projects/oscsimulator/
In the report, you can define 2 scenarios of running m processes given any parameters and any scheduling algorithm(s), and report the performance metrics PSS generates three text files in the output folder grouped in sub-folders named after the algorithm abbreviated name.

3) Does Producer/Consumer using Labs 4 and 5 count towards Task 3 requirements?

Yes and Lab Exercise 5 explained one way a 4 thread OS simulator can be designed.

- The first thread responsible for process creation is only a producer of the ready buffer
- The dispatcher (second thread) can do various roles:
    1. consumer to the ready buffer shared with the first thread,
    2. producer to the running buffer consumed by the CPU thread (third thread)
    3. producer as well to the I/O buffer shared with the I/O Thread (fourth thread).

- CPU thread is doing various roles as well
    1. consumer to the running buffer produced by the dispatcher, and
    2. producer to the terminated buffer,
- I/O thread is doing various roles as well
    - consumer to I/O buffer
    - producer to ready buffer

4) Does Deadlock management using Lab 6 simulator count towards Task 4 requirements?

Yes, and again it provides good example of simulating real processing. A process is again a line in the infile.txt, and resources are just arguments to the main method. More explanation to inspire your simulations follows:

When you run Lab 6 simulator, you type the command:

java TestHarness infile.txt 10 5 7

This tells the TestHarness main method, to know about the processes from the infile.txt, and read any number of following arguments, as the number available to the whole system. The TestHarness parses this input to understand that it has three resources, and there are 10 available of the first resource, 5 of the second resource, and 7 of the third resource. Since these are three numbers, then we have three resources, m = 3; and the three numbers from the arguments fills your available array.

In the infile.txt you then read the file to find a number of lines, each line represent a process. We have 5 lines, then n = 5.

Each line has the need for this process from each of the three resources until it terminates. This is why you are having three numbers. this fills your need matrix.

You start with allocated matrix initialised to zero, since as a process starts none of its instruction lines have requested or granted any resource yet. You simulate this requests, by typing in the console:

RQ 0 3 4 1

This will simulate a process 0 request to be allocated 3 of the first resource, 4 of the second resource, and 1 of the third resource.

RL 0 1 0 0

This simulates that process 0 will release 1 instance of the first resource, and zero of the second and third resource.

Any random simulation of requests and releases, while running the banker's algorithm to check if this request will leave the system in a safe state, will create a use case scenario that you can use in your final course work to explain the theoretical concepts in managing deadlocks in multi-instance resources systems. Otherwise, for single instance you can use the resource allocation graph as explained in the lectures and the slides and the book.

5) What if a resource link is not accessible anymore?

Public Domain is not owned by anyone. Authors have the right to remove links as they wish. Sites fail to exist sometimes temporarily and sometimes permanently. These are called resources. If some links exist fine. If other links are broken, may be you can research and find more on your own using the keywords in the module and the coursework. May be the book source code passed on in the labs and lectures are enough.
I added a Resources subsection to upload to SurreyLearn and resources I have. Currently it contains jsos.tar from the third resource that I might have downloaded before it was removed.
jsos
Resources do not need to be (and can not be) exhaustive list. I appreciate if you contribute to the discussion forum any resource you find on your own that other students can benefit from.

6) How far the user manual required in the report should be detailed?

It should be detailed enough to get your simulator running by a new user (Your assessor will follow the instruction you provide to grant the marks as rubric explains). A simple example is provided in PSS as follows:
https://github.com/MMayla/Process-Scheduling-Simulator/blob/master/MANUAL

7) How do I interface my code with an existing open source code?

This is a main objective in OS, being able to read the requirements of code done by other people, like the system developers, and prepare your code to call these methods/functions such as system calls.
This will require that you:
a) find out the method name (search manuals, reading code, reading textbooks, searching google),
b) find out the required arguments needed to be passed to it, how to prepare these from your data structures
c) and a way to interpret its output, and integrate that in your code.

Your code might be a simple interface that takes some input from the user (command line like in lab 2 simple shell example or GUI like PSS itself in lab 3) about the processes he/she wants to run. Then you form the arguments from the input and pass to the scheduling methods. Finally interpret the output to print out to the user. For example: the code in
https://github.com/MMayla/Process-Scheduling-Simulator/blob/master/app_pack/ProcessScheduler.java
uses the Process class to represent a process:
https://github.com/MMayla/Process-Scheduling-Simulator/blob/master/mini_pack/Process.java

You can use the same data structures, or simplify as you wish to cover the basic requirements that you want to simulate.

8) What 'Booting Steps' actually involves and how to simulate this?

Your OS manages a hardware system for you. It needs to identify this hardware while booting up (starting). In lab 2 additional material, booting steps of Raspberry Pi are explained. The main idea is that the OS need to identify the machine it is running on and start the initial process.
You need either to hard-code your assumptions by using Java constants or type these in an ini file (initialisation file such as the input file you used in PSS.jar) that is read by your simulator program (in the initial main method for example) to initialise the data structures required for your simulations. These parameters are at least
a) number of processors (this can be ignored and implied as one),
b) RAM size for the MMU,
c) Desk Size for the File Manager Subsystem,
d) what type of I/O hardware is attached to your system and available to your I/O subsystem.
I would come up with a file format that reads these parameters in the order described:
Example 1:
1
16 gb
500 gb
I: keyboard
I: Mouse
O: screen
O: printer
Example 2:
2
32 gb

1000 gb
I: keyboard
O: printer
Reading the first example ini file, will initialise one ready queue for one processor and one scheduler thread if you are using these options. Initialise the MMU to allocate no more than 16 gb memory and then blocks waiting on a condition variable (synchronisation) until a process deallocate memory. Initialise the desk to be of 500 gb for the file manager thread, and define a separate buffer for each input or output device if using buffers, or (direct memory access) will be studied more later on for the I/O subsystem.
Similarly, if you start your simulator reading the second example file, you will end up with different scenarios for testing as the number of threads will change, and execution states for the different test cases (processes to run) will vary.
I hope this clarifies a number of options that you can follow. This should not limit your creativity to design something different, simpler, more sophisticated, or as you wish.
Then you can call an init method (initialisation method that simulates the init first process) simulated as command line questions waiting for input from the user like you did in lab 2 in Simple Shell, or through starting a GUI such as the PSS in lab 3.

9) How do I start?
Because this is a final comprehensive coursework, some of the requirements are scheduled to be covered after the break and up to week 11. You can start with what you already know. A simple plan can be as follows:
As you studied in COM1028, you need analysis (Requirement Specification) which is given to you in the coursework requirements document with options. You need to read the rubric and in your report you explain what marks are you claiming based on your understanding and interpretation of the requirements. You have seen several options of implementing a requirement, you choose your set of options and explain them against each requirement. Based on the options that you will include in your simulator, start the design phase by drawing some Use cases, data flow diagrams, class diagrams, all that you think will help you plan your implementation and testing phases. The design phase contribute to your report marking.
Then, you can start implementation from lab 2 "Simple Shell" java source code for example. You will need incremental development of subsystems in a modular way, and get them to interface together.

https://blogs.oracle.com/java/introduction-to-modular-development

If you choose subsystem 1 (Process Scheduler), and a multitasking system, then instead of running each command you receive from the user straight away (Single Task system), try adding them to a ready queue and run a scheduling algorithms on them and using PSS.jar from Lab 3.

https://docs.oracle.com/javase/7/docs/api/java/util/Queue.html
You may choose to make your calling of scheduler to run whatever is currently available in the ready queue invoked by the user, or in a loop that runs every 10 seconds. You may choose to remain single task such as DOS, and create your test cases based on this. You might also exit the simulator and start again to run a new test case. These are all design questions that you need to plan and implement accordingly. You will need to explain your choices in your report.
If you choose to do Memory Manager, check Lab 8, will contain a virtual memory manager project and a sample model source code will be released on Week 9.
If you choose to do a File Manager, check Lab 9, will contain a file management OS subsystem, with a sample model source code will be released in Week 10.
If you choose to do an I/O Manager, check Lab 11, will contain some exercises on I/O procedures simulations that you can employ.
Back to the rubric while designing and implementing and check how you will score in each requirement and see if you want to include more to score higher. Think about the following:

- There are 10 marks on multitasking and multi-threading. You can score up to 3/10 by discussing these concepts theoretically in your report without any implementation.
- There are 10 marks on Synchronisation and deadlock prevention. Lab 4 and 5 will help you in Synchronisation and Lab 6 helps in deadlocks. See how you can benefit from these labs and the sample solutions presented.

You can also attempt any of the bonus exercises:

- Do a GUI for up to 10 marks:
  - https://docs.oracle.com/javase/tutorial/uiswing/index.html
- Do more subsystems for up to 25 marks as explained in the rubric of the subsystem attempted using the relevant labs or other designs of your choice or citing the original resource.

You can easily plan for a minimum of pass mark (40 %) in this coursework, and you can easily plan for more than 100 if you want to.

10) Is it required to connect all my simulations in one system with one interface (java file) or can it be separate java files invoked separately? So does there need to be a relation between a PSS and a memory management system or can they be stand alone?

The requirements specified did not mention one interface for everything. Operating Systems are usually a kernel image with basic minimal requirements to start the first process and identify the system it is running on. Then everything is a module that is loaded and unloaded as required. So you can choose how much manual work (required by the user or assessor) to test the requirements you choose to implement. It is not difficult to start one process that calls all other simulations you do. You can also attempt the bonus mark of a GUI where all your simulations are started from one interface. If you choose not to connect your simulations, it is alright, and the marks are assigned for each requirements in isolation as specified in the rubric.

If you do separate modules using sequential code only, this means your loosing the multiprocessing and multi-threading 10 marks and the synchronisation and deadlock prevention 10 marks. This leaves you with 80 marks to attempt by following the other requirements.

Revise Weeks 3, for multiprocessing communication (sockets) and Weeks 4, for Java multi-threading, 5 for Synchronisation constructs, and Week 6 for deadlock prevention

11) Do we use source code files from the existing systems, or do we call the executables from within our simulations?

For example if you are using PSS.jar, you can build an interface that uses pss.jar as an executable or include some of the source code files and call some of its methods. In both cases your simulations will require building 2 test cases (in this example 2 different input files for 2 different lists of processes and explaining the output of the scheduling metrics it produced in your report). Same applies to other subsystems and requirements.

12) I have been programming since last semester only, and not confident programming an OS simulator. How do I approach this coursework?

This FAQ document provide a lot of hints that can take you way above the pass mark. The interesting part of this coursework, that ***you can do simulations using existing systems and very minimal source code, down to no code at all.*** Writing a good report of test cases can take you way above the pass mark by showing good understanding of the theoretical concepts discussed, and provide test cases simulated using existing work.
A very modest plan can be as follows:

The report only is 25%, but if you run simulations using test cases of input and output this can collect marks for you from the subsystems marks as follows:
a- the booting you can get 1/5 for describing it properly in the report. Such as the assumptions your simulations are based on, the hardware specifications, and the instruction set that it can support, and any configuration parameters.
b- process scheduler can score up to 5/25 if properly defined in the report using simulations from lab week 3. Tracing two test cases by changing the input file twice and run the pss.jar and explain the output in your report will satisfy this requirement.
c- MMU can score up to 5/25 if properly defined in the report using the simulations from lab week 8. Tracing two test cases from the Input file and explaining how the input (logical addresses) to output (physical addresses) are produced will be satisfy this requirement.
d- file system can score up to 5/25 if you run simulations using lab week 9 lab in your report. You have 10 test cases included, running and explaining in the report 2 of them satisfies this requirement.
e- I/O system can score up to 5/25 if you run simulations and explain in your report.
This can take you with a good report up to 46 marks. *The 25 marks for the report are achieved if you fulfil all its requirements as outline in the main coursework document. Q28 below explains the user manual and technical manual requirements. If you do not develop your own code, the technical manual would explain the lab solution source code, whether the one provided on SurreyLearn, or another you got online. Being the one who wrote the code or not, is not important as long as you give proper references. You should be able to explain it in the report if you understood how it works, and managed to run the simulation test cases successfully.*
This is for those working for the minimum. You place yourself where you want to be. Working for perfection is always rewarded. Working for bonuses above the capped 100 (you can go up to 160/100, capped at 100 - two extra subsystems for 25 each, and 10 marks for the optional GUI) should be a rewarding educational experience that you can show case in your GitHub account (or similar) to target any career of your choice.

Remember that the internet and the Linux OS and many very successful projects were initially designed as an undergraduate student projects by students like you.

13) I'm unable to see how the concept of threads would be modelled using the PSS. From what I understand, each line of the input file represents a single process in the ready queue. So if each line is a process, what would a thread be?

PSS.jar simulated the processes sequentially and schedule them virtually by incrementing a time value representing clock ticks. According to the process requirements more ticks are consumed by one process, and according to the scheduling algorithm either a process is preempted after consuming its quantum, or continues once it's turn comes until it terminates. All is done sequentially without any true multiprocessing or multithreading.

There is no one design that is right and the rest is incorrect. In Lab 5, sample multithreaded OS simulator design is suggested. A second suggestion: you can elaborate on by making the scheduler is a thread, the CPU fetch decode execute cycle is another thread, file manager is a thread, memory manager is a thread, and I/O manager is another thread or any selection of these based on your choices.
A third suggestion: in a multicore/multiprocessor system you can have separate scheduler thread for each processor.
A fourth suggestion: You can have one thread per process execution that you start the thread and do not do scheduling yourself and let the Java Thread scheduler work for you, by managing the priorities of the created threads yourself only.
There is no end to how many different designs you can come up with as long as it's logical and you can explain in your report.

14) In lab 5 It was suggested to implement our own process file format as part of the coursework. Any more details on how do we approach this?

This is a fine grain level of process simulation that does not describe the process in terms of priority and expected CPU Burst only like PSS.jar. It also describe the process with all its instruction requirement that can navigate all OS subsystems that this module focused on. So the process becomes a real source code file containing some instructions. This will require that you come up with your Instruction Set that require memory allocation or addressing, File and I/O access, various computation and load and store commands.
You might benefit from checking this very nice low level simulator. It is the SIC/XE hypothetical computer from the Leland Beck's book System Software. The book is too much for year 1 CS - I would advise check the Java simulator only to keep it simple and correlate that with what you studied in COM1031. You can only see the assembly code in the images in this link :
http://jurem.github.io/SicTools/
and follow the assembly code as it executes and how the memory panel (right bottom window) changes values. So you can simulate memory as an array of fixed blocks, and every line of code loads/stores to a particular index a value as the logic require. You can also simplify this instead of Assembly Instruction Set, to something similar to Java Script that you can execute using the eval method:
https://docs.oracle.com/en/java/javase/13/scripting/java-scripting-api.html

15) if we implement the multi-threading/processing in only one subsystem, and then do the other subsystems without the use of threads, that is still alright?

Yes sure, once you demonstrate you can do multi-threading and synchronisation over one shared variable and explain it clearly in your report, this satisfies the requirement and no need to implement everywhere,

16) Which JDK version to we use?

The coursework is accepted in Java (any JDK), C/C++, and Python. Usually any higher version compiler is backward compatible with the previous versions. In your reports, specify exactly what is required to run your simulations. If special libraries are used, include their files (jars or so) in your source code submissions, or the commands to download them in an ubuntu machine where the testing is expected to take place. If you do not give clear instructions on how to your run your simulations, this will affect your marks.

17) Do we make threads representing running processes stop to simulate an actual swapping, or wait for long CPU bursts? how do we simulate processes execution time steps?

You do not want your simulations to be slow by stopping or making a thread wait to finish the quantum (if preemptive scheduling) or its CPU Burst (if non preemptive scheduling). PSS actually uses logical clock (just a counter starts at zero and keep incrementing as follows:
a) in preemptive scheduling algorithms, a quantum of 5 will make the counter gets incremented by 5,
b) a context switch in a preemptive scheduling will add to the counter a number of ticks as if this is wasted CPU cycles for swapping in and out processes, this can be a constant value that is hard coded in your source code, or initialised in an initialisation file
c) a CPU Burst of a process that runs until finished, in a non-preemptive algorithm, will add the complete CPU Burst to increment the counter as if it took that much, but all done instantly
d) Take care of not overloading the maximum value in the counter data type (overflow), for example restarting to zero when the maximum is reached, and count 1 cycle of overflows,
https://docs.oracle.com/javase/tutorial/java/nutsandbolts/datatypes.html
for example

```
if (clock >= Integer.MAX_VALUE - 1) {
```

```
clock = 0;
overflow ++;
}
```

18) how to test the variable speeds of when simulating multiple CPUs or cores?

If logical clock method is adopted as explained in Q 17 above, I can suggest a moderate CPU would increment the logical clock by the value of the CPU Burst of a process as suggested above, a double faster CPU, would increment the clock by half the process's requested CPU Burst as if it is running faster now, and similarly a slower CPU would increment the clock with a factor of how slow the CPU multiplied by the requested CPU Burst. If you are simulating multiple processors, you might have an initialisation parameter in the booting that give the scaling of the speed of a processor to decide the scaling factor.

19) How do we use the Raspberry Pi devices in the final coursework?

The Raspberry Pi was meant to be a case study of an OS that we downloaded its C and assembly language source code and compiled it in week 2. Last Year it was Android OS that we used as case study to focus on. You can read and understand the implementation details and extend through modules and system calls and recompile your new version. You can do this on any open source Linux flavour, including the educational Minix OS.
If you do not plan to work in this detailed level and want to work on Java simpler simulations like those we focused on in the labs, then any environment including the Raspberry Pi itself is a great development environment. Eclipse as an IDE provide excellent environment for faster writing of code, debugging and testing. You choose how you want to build your simulator.

20) I want full mark in the File System implementation, how do I go about it in comparison to the rubric?

The first column in the rubric where a student make no attempt at the requirement will be zero. The remaining columns are roughly 1/4 of the total mark for the assessment criteria. The marking will be both quantitative and qualitative. The total mark per section is maintained in the scaling, you could score 5/5 because you submitted the best work across all submitted, or less based on the quality of the work submitted to satisfy the requirement. There is also scaling in the university policy that could be applied to readjust the marks in case too high or too low marks are reported.

For the File System, there are 4 sections in the rubric, each roughly out of 5 marks subject to scaling if needed.
e: will score zero for no attempt.
d: Minimal Interface with the File System in Lab 9: as you see now, Lab 9 accepts typed commands by the user either in the command line, or in a test file script and pass it as an argument. Every run takes one file, or commands entered manually one after another. It should be clear that automating the input to the FS simulator will require writing source code to automate the creation of the commands using another interface. Doing this new interface again in the console is just to show understanding because it is fully implemented in lab 9 as test input files or interactive console. If you design GUI to visualise the disk usage for example such that you add another layer of simulation on top this will satisfy this category in the rubric and the 10 marks for GUI.
You can also remain in this category by running the test cases and explaining them very well in the report. This could give you up to 5 marks subject to the quality of the explanation in comparison to other students in the cohort.
c: Basic Interface with the FS Simulator and updated the Instruction Set to at least one of read, write, or seek: I have already suggested in Lab 5 a possible design to do so, by having an instruction set supported and a way to interpret a file or I/O requests from these instructions. The PSS.jar of week 3 does not provide an instruction set like executable code for the process. It only considers the CPU burst, priority and time of arrival to the ready queue. If you have updated this to have a file representing source code for each process. Then just like in your java programming you request a file access or input from user using keyboard and console screen, these are called I/O operations. You can simulate these as commands that the FS system of week 9 can execute. If you have not updated your PSS to include instructions to execute in every process, then it is difficult to integrate this with PSS, and you might need to come up with another idea. You remain in this category even if it is a separate instruction set for the File System that is not integrated with PSS.
b: Complete Interface with the FS Simulator and updated the Instruction Set to all read, write, and seek operations: like above, but here you completed all disk operations, or mainly these three operations, shut down and format is not really important in this assessment.
a: This category is for the innovators. If you update the source code your are given in lab 9 to a new design and explain your updates in the report. You should be able to see that achieving 25/25 in any subsystem require innovation not just step by step execution of lab sheet requirements.
All the above in this document, or in the lab sheets, or in any source code you find online, are design suggestions that should not limit your creativity to come up with something new. You should also accept the qualitative nature of the assessment and the scaling required by the university. You might be competing with other students who are committed to deliver the highest quality coursework, such that they raise the curve (the scaling), or accidentally a cohort of modest students that

lower the curve. You are in year 1 and perhaps neither you nor the teaching staff are sure about the cohort achievement levels, and this is why scaling will be essential. With the safety net policy you should work without pressure, but do your best anyway for the pleasure of the sense of achievement.

21) Similarly, a plan for a full mark in the Process Scheduling System implementation, how do I go about it in comparison to the rubric?

Same as question 20, the marks are both quantitative and qualitative. Scaling also is a must, and best/average submission will decide the scaling parameters. a plan to decide the categories would be as follows:
e: will score zero for no attempt.
d: **Defined in the report but did not implement, or implemented Single Process System such as DOS.** You would be in this category if the report contains the simulation of two test cases running the PSS.jar of lab 3 and the output are included and explained. Similarly if you implement something on top of Simple Shell of lab 2, for one task at a time will still be in this category.
c: **Defined in the Report and Source code submitted with Multitasking. Either not compiling, or basic errors in the logic:** In this category a genuine attempt of implementing a new PSS design using multitasking, but a range of compile or logical errors came up and made the testing not going as planned. PSS.jar is assuming one processor. Some ideas here could include more processor/cores each with its own dispatcher. Single processor can also require multi-threads as suggested in lab 5 using a separate process creation thread (such as Simple Shell but rather than executing, it would be adding to a ready queue), and separate scheduler thread that runs every few milliseconds to check the contents of the ready queue, and a CPU thread.
In any possible design in this subsystem or any in the coursework, you will need two test cases explained in the report, and it is alright to use source code from PSS.jar or any other online open source with proper referencing.
b: **Report and source code running including Multitasking but has some bugs:** at least everything is compiling and minimum 2 test cases are working properly. This category only allows for obvious logical errors that does not generalise to all possible test cases.
a: **Report and source code running as explained and includes good scheduling for multitasking:** This category is for the innovators. If you update the PSS you simulated in lab 3, to a new design and explain your updates in the report. A possible change to PSS is suggested in lab 5, which is to include a supported instruction set, and include a file for the instruction to be executed in each process. These instructions can as simple as math calculations, such that the CPU Burst would be a count of these instruction. This category as well is based on innovation and quality of work. Comming to this level will enable you to integrate the subsystems, by supporting memory allocation and freeing instructions to link with

MMU, File Operations to link with the File System, and I/O Operations to link with the I/O subsystem.

22) Similarly, a plan for a full mark in the Memory Management Unit System implementation, how do I go about it in comparison to the rubric?

Same as question 20, the marks are both quantitative and qualitative. Scaling also is a must, and best/average submission will decide the scaling parameters. a plan to decide the categories would be as follows:

e: will score zero for no attempt.

d: **Defined in the report but did not implement.** You would be in this category if the report contains the simulation of two test cases running the Virtual Memory Manager of lab 8 and the output are included and explained.

c: **Implemented basic MMU, with pages as blocks in the File manager simulator of Week 9. Some errors occurred:** Apologies the rubric said week 9, when it was supposed to be week 8. In this category a genuine attempt of implementing a new MMU design not the same one given to you in the lab, or significantly updating it, but a range of compile or logical errors came up and made the testing not going as planned. This means mapping logical to physical addresses as in Lab 8 is not a must here, and only allocation and reallocation with respect to main memory limits only.
Possible changes on top of what is given in lab 8, is as follows: if you integrate it in File manager simulations like Unix manages memory using a file system, and you can also use memory mapped files or any other idea,
I did not mention swapping with virtual memory up to category b. So, up to how far you implemented memory allocation and reallocation without errors you are in category c or b.

The only difference in not using virtual memory is not loading more pages than your main memory size. You would refuse to start a process or allocate more memory to a running process if your main memory is full, whether by returning it back to ready queue until the requested memory resource is available. Some systems might choose to terminate a process when it's requested process is not available.

In any possible design in this subsystem or any in the coursework, you will need two test cases explained in the report, and it is alright to use source code from logical to physical memory addresses mapping of Lab 8 or any other online open source with proper referencing.

b: **Implemented good MMU, with new data structure for pages:** This category will include significant change in one of the available online MMU design and source code such as lab 8, OSSim or others.

Possible changes are: i) making the logical space bigger than the physical space, ii) The logical to physical memory address mapping in lab 8 is assuming fixed size paging for example, you can update for segmentation of variable size, such that the logical to physical address translation is working well with segmentation variable length.

ii) or any change you think is logical in the data structures used.
Any change need to be reflected in the test cases.

a: **Implemented good MMU, with new data structure for pages and page replacement algorithms:** This category as well is for the innovators. Only in this category you will need virtual memory and swapping to load more memory pages than your memory size. In main memory only you do not need to replace an existing needed page for a new one. Lab 8 contains MMU for same size logical and physical main memory and a possible modification at the end of the lab sheet to include swapping to have more logical address space than physical, making this MMU actually Virtual memory manager. If you update the MMU in lab 8 to allow physical memory to be smaller than logical memory, and hence need a page replacement algorithm and keeping track of free pages, this will put you in this category. Another possible modification to place yourself in this category is to link the calls to MMU to an instruction set while running PSS. This integration can be innovative as you wish.
You will need good test cases to show that you filled up your memory pages and now replacing them to satisfy this requirement in full.

23) Similarly, a plan for a full mark in the I/O System implementation, how do I go about it in comparison to the rubric?

Same as question 20, the marks are both quantitative and qualitative. Scaling also is a must, and best/average submission will decide the scaling parameters. a plan to decide the categories would be as follows:
e: will score zero for no attempt.
d: **Defined in the report but did not implement.** You would be in this category if the report contains the simulation of two test cases running lab 11 and the output are included and explained.

c: **Implemented Basic I/O manager and updated the Instruction Set to call I/O functions :** Here you provide your own code for handling an I/O operation requested by a process. It will require an instruction set to be updated to specify the type of the

instruction set. This can be as easy as an instruction code and a flag to the supported I/O operation and the device number for example. A way to notify the process that the operation is completed. You can use lab 5 suggested design of using threads and notifications using shared variables of conditions, or monitor or wait and notify signals. If you use multi-threading you justify how good it is for the 10 marks, and proper synchronisation as proper at is for the 10 marks of synchronisation.

b: **Implemented complete I/O manager with 2 out of the I/O discussed services: scheduling, buffering, caching, spooling, device reservation, and error handling :** This category will include significant change in the basic simulations of lab 11. There is plenty of choices of which function to implement and provide test cases. You need only 2 of them and explain 2 test cases running them in the report. Scheduling is in Lab 10. Other concepts are discussed theoretically and some are spread through the labs. Buffering for example is in discussed in Consumer/Producer problem in the multi-threading labs. Caching such as lookup table for page hits, device reservations by locking techniques and others.

a: **Implemented complete I/O manager with 4 out of the I/O discussed services: scheduling, buffering, caching, spooling, device reservation, and error handling, with thorough discussion in the report :** Same as above, but with 4 operations implemented and tested with 2 test cases each in the report. Without 4 operations, a proper integration to the other subsystems you implemented to have one OS Simulator doing everything, will position you in this category as well. Such as one instruction set that is supporting the 2 subsystems you chose: Memory Operations, File and I/O operations, and proper scheduling and management of queues.
In the last 4 questions, there are plenty of design opportunity for including Multi-threading (10 marks), Synchronisation (10 marks), and GUI (bonus 10 marks).

24) How to incorporate disc scheduling of lab 10 algorithms into the file system of lab 9?

The file system in lap 9, executes all disk operations once received one by one. You can add these requests to a queue or array, and execute them all after scheduling them using the simulation used in lab 10.

25) If the subsystems need to be linked how do I link a process scheduler and memory management unit together?

The only way I can think of is to design an instruction set that says what commands need to be executed in a process to complete it. These instructions can be a memory allocation instruction, a memory de-allocation instruction. To

generalise the question to other subsystems linking, you can link with file and I/O subsystem, you will need instructions for opening a file, writing, reading, closing, and I/O operations instructions.
Another simulation suggestion, is to expand the input file used in PSS.jar to include columns about memory allocations requests, file and I/O requests, You can ignore the timing of these requests to simplify these simulations (such as early at the start of the process or in the middle or at execution paths that might not happen). You can also include a column for when these requests are needed as if only one execution path is available. Such as having the column of memory allocation be followed by logical clock count of when this request is made. if the CPU Burst of the process is 100, you can say at clock 20, memory allocation of that much, at clock 40 file read, and clock 60 file write, at clock 70 memory de-allocation, at clock 90 I/O request from this device, and arrange the columns as you want and explain your design. As you see you might need more column to compete all input required, such as which file, which I/O device, how much memory and so on.
Many design options are available and you can think of an easiest way to do things, and as you test and have more time, you can add more features as you wish.

26) For section 2bii of the source code requirements - I understand what cache is, however I don't understand how to implement it and make it useful for this coursework. Could you please provide some examples to assist me?

The second textbook has a python code example to simulate a cache model. It is available in this link:
https://github.com/wimvanderbauwhede/OSH/blob/master/Code/Ch1-memory-centric-system-model/cache-model/cache_model.py
It is like the lookup tables for memory pages, but it is memory hardware that is placed closer to the CPU so reading/writing to them is faster than reading/writing to main memory pages. So a page is copied from main memory to L1 (closest to the CPU) when it is needed. Based on the size of L1 and the size of the logical page, you will be having a number of pages available for faster access. When it is full, you have page miss, and will start copying from main memory to L1 by evicting a page using one of the page replacement algorithms discussed. Of course you know that main memory might have the page and another layer of page miss might happen to read from swap virtual memory. Many computers support L2 as well, and even L3 shared between different cores. Think of these as the hierarchical lookup tables in software, but they are in hardware, connected by buses that has variable latency for each level and its proximity to the CPU or core. In simulation, you do everything in software and have the freedom to choose from the simplest models, no cache at all, to one level of cache or more, based on how sophisticated you want your simulations to be.

27) I am confused about handling the PCBs. Do we need to create virtual hardware? Our own architecture? Or can we just let Java allocate the memory for us?

Hardware is simulated as Disc in Lab 9 File system simulator. Other hardware devices can be just data structures and buffers. Also Memory hardware in Lab 8 is nothing but data structure not hardware. If you let Java allocate memory for you, then you are not simulating memory management unit. Any way in the simulations you are allocating a lot of memory of these needed data structures and it is Java and the OS underneath is doing this for you, but if you are simulating MMU, you need to follow some design like that used in lab 8.

28) Could you explain what exactly do you want in "User Manual" and "Technical Manual"?

A user manual is like your router manual. It says the steps to operate in clear language without missing out on a step that makes the software or the hardware not easy to use. The user is not interested in how you built it, he is interested in how to use it. This is the what to do.
A technical manual is a developer talking to a developer. You Explain to another developer who knows the technical details: class diagram, use cases, data flow, data structures, algorithms that another developer need to understand to develop new features on top of your work or fix bugs or change it's behavior. This is The How to do what the user is just executing.

29) How to explain in the report the fourth requirement: The architecture of the CPU and the board and peripherals that your operating system is assuming and how it can be compatible with other architectures if possible, or why not.

This is related to every subsystem is isolation. If you built single task system (one task at a time and finish before starting another) in the process scheduling , then you are Having one CPU or core, if you are using multitasking but not in parallel, then still one CPU/Core. If you are using different threads and dispatchers per CPU, then you can either be homogenous or heterogeneous structure, explain how you assumed the multicore or multiple processors architecture is

In the MMU, the memory addressing and space will require some sizes and constants or initialization or assumptions, you need to discuss that. Same logical space as physical, more and using swaps, paging or segmentation, different cache levels or not. All these are assumed on the hardware and simulated using data

structures or hypothetical commands that needs interpretation from your side

Similarly file and I/O subsystems if use them, there will be assumptions about block or character types, sequential or random access, memory mapped files or not, DMA or not, various assumptions on the hardware that makes the OS behave differently on the lower levels

30) The rubric includes 10 marks for multi-processing as its own section. However in the mark bands for the process scheduler it also states that we cannot go past the first mark band if it is not multiprocessor. Can you explain if there is overlap in the requirements and marks distribution?

You can develop one task system at a time such as MS Dos, and you can develop a concurrent multi-process but not in parallel, such as the PSS.jar is doing. PSS.jar is accepting several processes and scheduling them by dividing the CPU time between processes, but no two processes are running at the same time simultaneously. This is for the marks for process scheduling. PSS.jar is sequential, no threads working together or processes working together, and no shared variables to protect with synchronisation.
For the multiprocessing section it is actually for the parallel processing supporting multi-core or multi-threading. This is by creating threads with shared variables or processes that communicate. These will be running simultaneously at the same time taking advantage of the multi-cores on your systems.

31) I do not understand how to create test cases to write about in the report?

Every lab almost had test cases. The input file to pss.jar in week 3, is one test case scenario. You need to design another input file with different number of processes and requirements of CPU burst, priority, and time of arrival to have another test case to explain the produced chart and metrics in the output folder. One algorithm is one test case. Also reduce screen shots, and use text copy/paste where possible. The produced chart need to be an image if explaining its output.
The deadlock prevention algorithm in week 6 had commands that you can arrange to create a scenario of resources requests and releases that you can create a test case from and show when the algorithm decides a safe state and grant a request and when it denies. Two scenarios would satisfy the requirements.
The memory management lab in week 8 has an input file of Logical addresses as test cases, to show the resulting physical addresses and confirms that the mapping was correct according to the logic of the algorithm.
The file system of week 9 has commands and a test folder full of test scenarios. You can use any 2 of these pre-designed test cases or come up with your own.

The I/O subsystem of week 11 has a test case of adding I/O devices and receiving I/O requests and processing them. There is only 1 test case predefined for you, you need to come up with another following the same logic.
All these are examples of test cases if you have used the code provided for you in the labs. If you have used other systems you downloaded from online resources, or came up with your own solutions, you can only use these ideas to guide how you can create other suitable test cases to your design and show case how the simulations works and prove the correctness of the logic implemented.

32) Can you explain further the multithreading and synchronization mark distribution?

For the multithreading :

e: will score zero for no attempt.
d:Explained in the report only
c. Attempted in source code with compilation errors
b. No compilation errors but has logical bugs, such as still sequential or not completely parallel as it should based on the logic you are attempting
a. Good parallelism. Please revise the lab to make sure they are indeed parallel to be in this band

For synchronization:

e: will score zero for no attempt.
d:Explained in the report only or placed your synchronization constructs in the wrong place in your code.
c. Using deadlock free library
b. Like c and also justified this in the report.

a. You did the resource allocation graph or banker algorithm yourself embedded with your code.

33) how do I reference the labs source code?

Lab 1 ,5, 7, 11, are designed for this module and you can use something around the following IEEE citation updated for each lab:

M. Helal, “SurreyLearn COM1032 19-20 Offering,” Introduction to Some OS Functionalities Commands in Linux & Raspberry Pi 3, Feb. 2020. https://surreylearn.surrey.ac.uk/d2l/le/content/188676/Home?itemIdentifier=TOC.

Lab 3: cite PSS jar following coursework instructions on citing source code in general.

Lab 2, 4, 6, 8, 9, 10: cite the first text book, such as the following or specific lab title:

A. Silberschatz, P. B. Galvin, and G. Gagne, Operating system concepts with Java, 8th ed. Hoboken, N.J: John Wiley & Sons, 2010.

For other online resources, search online where you downloaded the source code, if there is a paper accompanying the source code, you reference the paper. Otherwise, you can reference as a blog website that you accessed using the following format as an example. Other styles work as well:

Template:

[1]Author Initial. Author Surname, 'Title', *Publication Title*, Year Published. url .

Example:

[1]d. dinaen, 'fair', *fair*, 1981, http://wwww.blog.com .

In-text citation

Place this part right after the quote or reference to the source in your assignment.

Template

[1]

34) I am using the lab solution for some subsystems and updating other. For the ones I am using from the labs, I am running two test cases and explaining them. Do I keep the files of the solution you gave us on the eclipse project or remove them?

You need to keep your submission size as small as possible to make things easier for us. Include only what you changed, and reference the lab contents as shown in the previous question.