Operating Systems

6CM503

Computer GE 645 with OS Multix and its creators around

01 Assessment Brief 2025/26

Linux Command Line Interface (CLI): Shell Script Design

Dmitry A. Zaitsev

1. Module Leader

- Dr. Dmitry A. Zaitsev, V. Markos (Athens), A. Tziola (Thessaloniki)
- Email: d.zaitsev@derby.ac.uk, Phone : 01332592678
- Office: MS229. Recommended days for individual communication: Monday, Wednesday.
- Website: https://www.derby.ac.uk/staff/dmitry-zaitsev

2. Key dates and details

Assessment Detail	Assessment Information
Assessment Type:	Individual Report, Program Code
Assessment Weighting:	20 %
Word count/Length:	1,000 words / 3-4 pages (including listings)
Learning Outcomes:	1, 2
Submission Method:	Blackboard Assignment
Submission Date:	17:00 Greece Time, 28/11/2025
Provisional Feedback Release Date:	17:00 Greece Time, 19/12/2025

3. Description of the assessment

User interface is one of the basic interfaces of OS, together with interfaces with running processes and computer hardware. As a standard de facto, a general-purpose modern OS provides two kinds of user interface: Graphical User Interface (GUI) and Command Line Interface (CLI). GUI and CLI are mutually complementary. Besides, CLI is laconic and flexible; sequences of commands are composed into pipes and scripts that bring the functionality of a programming language. Scripts play a significant role in configuring OS, especially during booting and shutdown processes, as well as in configuring and installing application software.

The goal of the assessment is to master the command line interface of Linux and the Shell programming language to compose scripts, which play a significant role in the tuning of Linux and the solving of application tasks. Brief and powerful CLI allows us to administer Linux efficiently even via low-speed emergency remote connections.

With this assessment, students master skills of composing systems scripts in shell (bash) language based on the incremental learning approach detailed in the Laboratory and Home Training. They start composing sequences of commands to implement required actions to learn bash commands for working with files, processes, pipes, and filters. Finally, students compose a single but not elementary pipe, represented as a small script and a systems script for monitoring OS resources. For scripts, they have 12 tasks assigned individually within an explicit list of students.

Basic information:

- Lectures 1-5
- Laboratory and Home Training, section 1, weeks 1-6
- OS-individual-task-number-assignment

4. Relationship to Programme Assessment Strategy

The first task of the module Coursework four portfolio tasks.

5. Attributes and Skills

The attributes and skills are listed in the following table.

	Skills	Links to useful resources
☒	Critical thinking	Analysis of historical and contemporary OSs.
☒	Communication	Working within common Linux server environment.
☒	Collaboration	Partial teamwork on the same task variants.
☒	Creative problem solving	Script, utility, and kernel loadable module design
☒	Self-direction & planning	Planning efforts and activities to satisfy task submission deadlines.
☒	Numeracy, statistics & financial literacy	Top X lists and statistics on most widespread OSs for specific application domains, awareness of financial aspects with OSs' comparative cost and reliability
☒	Digital	Students not only use but create digital technologies
☒	Resilience	Students learn how to avoid stress inflicted by with necessity to satisfy deadlines and recover after busy times
☒	Adaptability	Adaptation to using rather novel, for students, type of OS compared to officially supported by UoD MS Windows
☒	Leadership & future thinking	Developing kernel loadable modules, students think about OS of future

6. Assessment Content

Sharing basic resources in multiuser systems leads to problems with involuntary or intentional use of significant amounts of resources by a single user or a group of users, blocking the entire system

work. It affects the system security and robustness. While the general solution is provided by additional facilities for resource quotas, system scripts, which monitor the current OS state and ongoing changes, help in understanding problems and finding best solutions to resolve them.

The central task of the present assessment is the system script design for monitoring OS resources according to the student's individual task.

The assessment involves an incremental learning approach facilitating a better student experience specified in the corresponding section of the Laboratory and Home Training. We start with CLI training, learning commands and their options, then compose a single-line script in the form of a pipe, and finally design a system script for monitoring OS resources.

6.1. Brief specification of the assessment workflow

Part 1. Design a pipe

Design a pipe that extends the functionality of Linux as an additional command with the specified name and function, and test the pipe work.

Recommended workflow:

1. Draft a sequence of required commands.
2. Find out required options.
3. Test each command separately.
4. Incremental design adding commands to the pipe.
5. Compose and test the final pipe.
6. Save the pipe in a file, add parameters.

Individual tasks: Section 6.2.

Part 2. Design a script

Design a program (script) in Shell (bash) language according to a given individual task and test the script work.

Recommended workflow:

1. Draft a sequence of required commands.
2. Find out required options.
3. Test each command separately.
4. Save the script in a file, add parameters.
5. Incremental design, adding commands to the script.
6. Add control flow to the script: branching and loops.
7. Compose and test the final script.

Individual tasks: Section 6.3.

Recommendations for pipe and script design: Laboratory and Home Training, section 1.

Content of report: Section 6.4.

6.2. Individual tasks for pipe design

1) lx – list of executable files of a specified directory.
2) pu – list of processes started by a specified user.
3) nx – number of executable files in a specified directory.
4) npu – number of processes started by a specified user.
5) np – number of processes running a specified file.
6) mp – list of users who started a specified (or greater) number of processes.
7) kn – delete all processes running a specified command.
8) bp – print information on a specified number of processes having the greatest time of running on processor.
9) bf – print information on a specified number of files having the gratest size.
10) kp – delete all stopped processes of a specified user.
11) ml – print information on a specified number of files having the greatest number of links.
12) ll – list of users who are owners of files in a specified directory.

- **Individual task number is pre-assigned according to the file of individual task numbers (OS-individual-task-number-assignment.pdf); please use column 2, choose the row based on the student name and surname, specified in column 1**

6.3. Individual tasks for script design

1. Track users who started more than specified number of processes.
2. Track changing the number of file links in the specified sub-tree of directories.
3. Track users who started more than specified number of processes from one terminal.
4. Track changing access rights in the specified sub-tree of directories.
5. Track creating new files/directories in the specified sub-tree of directories.
6. Track changing free disk space for a specified disk.
7. Track changing access rights in in the specified sub-tree of directories.
8. Track login/logout of users into the system from specified terminal printing the time of user work.
9. Track deletion of files/directories in the specified sub-tree of directories.
10. Track changing processes priorities in specified range.
11. Track the list of terminals attached to processes with specified state.
12. Track processes which used the amount of processor time exceeding the specified value.

- **Individual task number is pre-assigned according to the file of individual task numbers (OS-individual-task-number-assignment.pdf); please use column 2, choose the row based on the student name and surname, specified in column 1.**

6.4. Content of the assessment report

1. Pipe specification
   1.1. How to use the pipe

1.2. Pipe description
1.3. Brief description of used commands
1.4. Results of the pipe testing (no less than 3 tests)
2. Script specification
2.1. How to use the script, working files specification
2.2. Script description supplied with graphical schemata
2.3. Brief description of used commands
2.4. Results of the script testing (no less than 3 tests)

The report should be supplied with an archive containing (in textual files ready to run) the pipe, and script supplied with comments on how to use them.

7. Assessment Rubric

Criteria	Excellent (70-100%)	Very good (60-69%)	Good (50-59%)	Satisfactory (40-49%)	Unsatisfactory (<40%)
Design a pipe, 30%	Successful implementation of a given pipe	Successful implementation of a given pipe with minor imperfections	Successful implementation of basic functions of a given pipe	Implementation of basic functions of a given pipe with some errors	Basic functions of a given pipe have not been implemented
Design a script, 50%	Successful implementation of a given script	Successful implementation of a given script with minor imperfections	Successful implementation of basic functions of a given script	Implementation of basic functions of a given script with some errors	Basic functions of a given script have not been implemented
Report writing, 20%	The report is exceptionally well-structured, clearly written, and professionally presented.	The report is well-structured and clearly written with minor errors.	The report is complete but may lack clarity or detail in some sections.	Report is poorly organized or lacks clarity. Analysis is weak or incomplete.	Report is disorganized, difficult to follow, and lacks key elements.

8. Assessment AI-assistance

In this assessment AI-assistance is permitted in the following ways:

- Installation troubleshooting
- Check of obtained results

In this assessment AI-assistance should not be used for:

- Code composition
- Writing report

It is YOUR RESPONSIBILITY to check ALL information generated by generative AI tools. Any misuse of generative AI tools could be considered ethical academic misconduct (as per Academic Regulations, Academic Misconduct, Section J2). If in doubt, please consult your module leader.

9. Anonymous Marking

You must submit your work using your **student number** to identify yourself, not your name. You must not use your name in the text of the work at any point. When you submit your work in Turnitin you must submit your student number within the assignment document and in the *Submission title* field in Turnitin. Guidance is available showing how to do this.

10. Assessment Regulations

The University's regulations, policies and procedures for students define the framework within which teaching and assessment are conducted. Please make sure you are familiar with these regulations, policies and procedures.