University of Roehampton London

School of Arts, Humanities and Social Sciences

Module title and code: **Computer Systems CMP020L002S**

Title of coursework: **Assembly Language Programming**.

Learning outcomes	The coursework evaluates the following learning outcomes of the module for writing assembly code segments by doing the following: • LO2: Develop state machine descriptions for problem statement • LO3: Perform simple computing calculations using assembly language and test your ability to use assembly instruction commands to build your program.
Assessment weighting	30%
Maximum mark	100
Submission details (e.g. submission link)	Moodle site for Computer Systems CMP020L002S
Word limit (if applicable)	--
Date set	13/02/2025
Deadline	05/03/2025
Feedback and marks	Submission via Moodle and Demo
Assessment setter's name	Jose Paredes

Academic Misconduct:

"Academic integrity and honesty are fundamental to the academic work you produce at the University of Roehampton. You are expected to complete coursework which is your own and which is referenced appropriately. The university has in place measures to detect academic dishonesty in all its forms. If you are found to be cheating or attempting to gain an unfair advantage over other students in any way, this is considered academic misconduct, and you will be penalised accordingly."

Further details about "Student Code of Conduct" and "Disciplinary Regulations" can be found at:

https://www.roehampton.ac.uk/corporate-information/policies/

Assessment Introduction

The coursework is worth 30% of the total module marks. It is a group project in which the students will apply foundational assembly language concepts to solve practical problems related to their **Computer Systems** course. The tasks are designed to enhance the understanding of arithmetic operations, loops, and array manipulation by working with real-life scenarios, such as student group assignments, enrolment calculations, and department statistics. Each question builds on the concepts taught in the class, enabling students to implement logic-based solutions. By completing this assessment, students will improve their coding proficiency and develop problem-solving skills.

Scenario

A new course module, Computer Systems, has been introduced for a cohort of 290 students between the ages of 19 and 42, studying in undergraduate (UG) and postgraduate (PG) computing programs.

- Out of the total, 75 are postgraduate students, and 15 of these PG students chose not to participate in the practical labs.
- The UG program comprises four departments: Computer Science (98 students), Cyber Security (59 students), Software Engineering (43 students) and Web Design (15 students).

The students are divided into five project groups: PG-A, Computer Science-B, Cyber Security-C, Software Engineering-D, Web Design-E and mixed group-F.

The module reached the point where students were split into smaller teams to work on lab simulations and assembly language exercises. The number of students allowed in a team is 3. Students have a week to choose their partners for mini-groups, and any remaining students were automatically assigned.

Finally, within the last three days, 42 more students joined new groups before the final deadline.

Write assembly language programs using NASM to perform the following functions:

1. Write an assembly language program to display a **message introducing yourself** to your chosen mini group.

2. Write an assembly language program that categorizes a given age into different life stages: "Young Adult", "Adult", "Mature Adult" as follows:

- **Young Adult:** 19-25 years, **Adult:** 26-35 years, **Mature Adult:** 36-42 years

The program *should prompt the user* to enter an age and display the corresponding life stage.

3. Write an assembly language program that will subtract the number of PG students from the total number of students in the module and display the output as **"Total number of UG students: x"**, where x should be the actual number of UG students.

4. Write an assembly language program to calculate **how many mini groups** there are in each of the five project groups: A, B, C, D, E and F.

5. According to the latest statistics, there is a 12% decrease in the number of female students compared to last year. If 60 female students are enrolled this year, write an assembly program to calculate **how many female students** were enrolled last year.

6. Draw a state machine for question 2 and another on for question 5.

Submission Requirements

Everything must be submitted within the due date to be considered for a complete submission for this assessment. **The submission must be done by only 1 member of each group.** Demo will only be held after the work has been submitted through the specified link.

Your submission must consist of five ".asm" files, one for each Assembly program, and a PDF or image document for the state machines. All assembly files must be appropriately commented in each file using the proper Assembly syntax.

Your Assembly code must be buildable using NASM. This is not negotiable. If you use a different Assembler, your assessment will be graded at 0.

Do not submit any other files. Your mark will be adversely affected if you submit incorrect files.

Assessment Rubric

Performance Indicator	**Excellent**	**Very Good**	**Adequate**	**Inadequate**
Code Functionality and Accuracy	All programs function as expected, producing correct results for all inputs and meeting all specifications.	Most programs function correctly with minor errors in output for a few inputs. Meets most specifications.	Some programs function, but several do not meet specifications, with errors in output or missing functionality.	Programs do not function or produce incorrect results for most inputs. Fails to meet core specifications.
Use of Assembly Language Instructions	Correct and effective use of assembly instructions, showcasing a deep	Mostly correct use of assembly instructions, with minor inefficiencies or mistakes.	Partial understanding of NASM, with several incorrect or inefficient uses	Little to no correct use of assembly instructions. Demonstrates poor or no

	understanding of the NASM assembler and instruction set.	Demonstrates a good understanding of NASM.	of assembly instructions.	understanding of NASM.
Problem-Solving and Logical Structure	Excellent logical structure in the code, with well-thought-out solutions and clear implementation of problem-solving techniques.	Good logical structure, with minor issues in implementation or approach, but overall sound problem-solving techniques.	Some problems solved correctly, but with significant issues in logic or approach in others. Solutions may be unclear or incorrect in parts.	Poor problem-solving skills, with little or no logical structure to the code. Solutions are unclear or incorrect in most parts.
Code Efficiency and Optimization	Code is highly optimized and efficient, using minimal resources while achieving the required functionality.	Code is reasonably optimized, with some areas for improvement in resource usage.	Code is somewhat inefficient, with noticeable areas where optimization is lacking.	Code is highly inefficient, with poor resource usage and little to no optimization.
State Machine design	All outputs are presented clearly, with well-structured state machines and flowcharts for relevant problems.	Most outputs are presented clearly, with minor issues in the presentation of state machines or flowcharts.	Output is unclear or incomplete, with several issues in state machine or flowchart representation.	Little to no output presented, or output is largely incorrect. State machines or flowcharts are missing or poorly presented.
Defence of Work (demo)	Demonstrates a thorough understanding of the project, answers questions confidently and accurately, and provides insightful explanations.	Shows a good understanding of the project, answers most questions accurately, and provides clear explanations.	Demonstrates a basic understanding of the project, answers some questions accurately, but explanations lack depth.	Shows limited understanding of the project, struggles to answer questions, and provides unclear or incomplete explanations.