University of Glasgow

SAMPLE EXAMINATION
(2 hours)

DEGREES OF MSc, MSci, MEng, BEng, BSc, MA and MA (Social Sciences)

ADVANCED SYSTEMS PROGRAMMING (M)

Answer 3 out of 4 questions

This examination paper is worth a total of 60 marks.

The use of calculators is not permitted in this examination.

INSTRUCTIONS TO INVIGILATORS: Please collect all exam question papers and exam answer scripts and retain for school to collect. Candidates must not remove exam question papers.

1. **(a)** A running program will reference memory on the stack and on the heap. Outline what data is stored on the stack and what data is stored on the heap. Explain why it is necessary to store the stack and the heap in separate regions of the virtual address space. [6]

(b) One approach to automatic management of heap memory uses reference counts attached to each object. The runtime increments the reference count when a new reference to the object is created, and decrements it when a reference is removed. The memory allocated to an object is reclaimed when the reference count of the object is decremented to zero. Briefly outline the main benefits and problems inherent in using reference counting as a means of automatic memory management. [4]

(c) The Rust programming languages tracks data lifetimes and uses region-based memory management to automatically manage memory. With reference to lifetimes, ownership, borrowing, and the different types of references, explain how Rust manages memory. [10]

2. **(a)** Modern systems programming languages are increasingly focussing on how to provide effective support for concurrency. Discuss how the data ownership rules in Rust help avoid data races due to concurrency. [6]

(b) The Rust and Erlang programming languages both support message passing via channels to communicate data between threads. Channels in Rust are statically typed and the compiler will check that the messages exchanged via a channel conform to the stated type. Erlang is dynamically typed and any type of message can be sent to any thread. Discuss the trade-off between the two approaches. State which do you think is best for systems programming, and explain why. [6]

(c) The increased focus on concurrency in programming languages is due to the increasing amount of concurrency present in microprocessors and systems. With reference to Moore's law and Dennard Scaling, explain why processors are moving to designs with increasing numbers of cores. [8]

3. **(a)** Describe how types and functions can be used to model state machines using a language, such as Rust, that tracks data ownership. Explain how this helps to manage state variables and resources. Discuss why this helps to achieve more robust software. [8]

(b) State machines, especially those used to model network protocols, can also be represented using asynchronous code and coroutines. Discuss why asynchronous functions that can await intermediate results are considered desirable for network programming. [6]

(c) Discuss the extent to which you believe asynchronous programming succeeds in its goal of simplifying implementation of concurrent state machines for network code. [6]

4. **(a)** The recommended reading for lecture 8 included the paper on "The bugs we have to kill" by Bratus, Patterson, and Shubina (USENIX ;login:, August 2015). This paper suggests that the way we build networked applications must change if we are to reduce security vulnerabilities. The authors suggest that we should move to using strongly typed and memory safe programming languages to improve overall robustness of the code, along with formally-specified grammars and parser generator tools to ensure safe parsing of untrusted data. We also discussed these ideas in the lecture, where it was suggested that such approaches won't eliminate security vulnerabilities but, if used carefully, they might be able eliminate certain common classes of vulnerability, and make others less likely by making hidden assumptions – and their security consequences – visible.

Do you agree with the thesis of this paper and the lecture discussion? Discuss the extent to which you believe that changing the programming language and using automated parser generator tools will help to address the challenges in building secure networked systems. Outline what programming language, runtime, and parsing features you consider important for supporting secure systems programming, and what are harmful, giving examples to illustrate key points. [20]