211 — Operating Systems – Tutorial
Introduction

Peter Pietzuch <prp@doc.ic.ac.uk>

Note that the solution notes below only briefly list (some of) the key points that should be included in an answer. They are by no means complete. In an exam, you are expected to spell out the solution more fully and include a detailed explanation of your reasoning.

1. The issue of *resource allocation* shows up in different forms in different types of operating systems. List the most important resources that must be managed by an operating system in the following settings:

   (a) Supercomputer

   Solution Notes: Processors, memory — any resource that affects the computational performance of the supercomputer.

   (b) Workstations connected to servers via a network

   Solution Notes: The network is a resource that is shared by all connected computers. Access to it may be managed by the OS through the network protocol stack that avoids congestion using protocols such as TCP.

   (c) Smartphone

   Solution Notes: A smartphone may have limited battery life, so battery power is a resource that has to conserved by the OS. For example, the OS may decide to put the CPU to sleep or switch of certain unused hardware features.

2. What is the *kernel* of an operating system?

   Solution Notes: The part of the operating systems that is always in memory and implements the most commonly executed functions of the OS. The OS kernel executes in kernel or privileged mode and therefore has complete access to all hardware (in contrast to user-mode processes).

3. Why is the separation into a user mode and a kernel mode considered good operating system design?

   Solution Notes: A process executing in user-mode can cause less "damage" because it is restricted from executing privileged operations. By splitting an OS design into user-mode and kernel-mode components, the OS designer makes it explicit which parts of the OS require raised privilege and full access to the hardware. This is an example of the "principle of least privilege" that should guide the design of any secure systems.

   Give an example in which the execution of a user processes switches from user mode to kernel mode, and then back to user mode again.

   Solution Notes: A system call (using a trap instruction) is an example of this.

4. Which of the following instructions should only be allowed in kernel mode, and why?

   (a) Disable all interrupts

   (b) Read the time of day clock

   (c) Change the memory map

(d) Set the time of day

Solution Notes: (a), (c) and (d)

5. A *portable* operating system is one that can be ported from one system architecture to another with little modification. Explain why it is infeasible to build an operating system that is portable without any modification.

   Solution Notes: By definition an OS has to interact with the hardware directly. It therefore contains code that depends on the specifics of the processor architecture such as its instruction set and the memory management system.

   Describe two general parts that you can find in an operating system that has been designed to be highly portable.

   Solution Notes: A portable operating system can be split into a (1) platform specific and into a (2) platform independent part.

   The platform-specific part contains any OS code that is dependent on the given processing architecture and platform. All other OS functionality is implemented as part of the platform independent part that uses the API exposed by the platform-specific part to invoke any low-level hardware functionality. When porting the OS, only the platform-specific part has to be reimplemented for a new platform. The rest can just be recompiled for the new architecture.

   This type of division can be found in the Linux kernel, which makes it (relatively) easy to port it to new platforms.