**Score: \_\_\_\_\_**

**OSA1 – Operating System Abstractions**

**Activities**

COMP256 – Computing Abstractions

Dickinson College

Spring 2023

Prof. Grant Braught

**Name:**

**Introduction:**

Today’s class introduced the Operating System Abstractions topic. It took stock of what we have learned so far and identified the key ideas from earlier units that will be essential to keep in mind as we learn about operating systems. Several definitions were given for an operating system and some of the chasms of complexity that needed to be bridged were highlighted. Finally, the metaphor of a college was used to get a feel for the key Operating System Abstractions that we will be studying. Today’s activities will give you a better feel for what an OS is, ask you to develop your own metaphor and dig into how the OS actually gets started (i.e. how your computer boots up.)

**Defining an Operating System:**

Today’s class gave two definitions of an operating system adapted from a popular operating system’s textbook and a tech web site. But there are about as many different definitions of an operating system as there are people defining them.

🔑 1. Using your favorite search engine find and read some definitions of an operating system until you find one (different from those given in class) that describes an operating system in a way that makes sense and agrees with the way you think about operating systems as this point in time. Give that definition here and indicate where you found it.

🔑 2. Pick one of the definitions of an operating system from class. In a few sentences of your own words, compare and contrast the definition you found with the one from class that you picked. What ideas do both address? What ideas does one address that the other does not?

🔑 3. The first definition of an operating system given in class concludes that the operating system creates “an environment in which programs can be executed in a convenient and efficient manner.” Now, recall that a basic stored program computer (one CPU with one core) can only execute one instruction at a time. Thus, on such a machine if the OS is running, then a program is not. Similarly, if a program is running, then neither the OS nor any other program is running. With those limitations in mind, answer the following questions:

a. How does your OS make it convenient to execute programs?

b. For your OS to produce those conveniences requires that the machine spend time executing instructions that are part of the OS. When these instructions from the OS are executing are instructions from your application programs executing? Briefly explain your answer.

c. Given your answers to parts a and b, it should be apparent that there must be a tradeoff between user convenience and the efficiency with which application programs are executed. To increase user convenience, we must decrease the efficiency of user programs or vice versa. Summarize this tradeoff and why it exists in a few sentences of your own words.

🏆 d. There are lots of different operating systems for lots of different types of computing system. These range from high-performance computers used for computationally intense tasks like environmental and weather modeling to those used on consumer-focused devices such as “smart” phones and watches. Briefly explain why different OSs for different devices might choose a different balance between convenience and efficiency.

**Metaphors for Operating Systems:**

Today’s class introduced a College as a metaphor for understanding the role played by an operating system. This section will explore that metaphor a little more and then ask you to create a metaphor of your own.

🔑 4. In the college metaphor what/who plays the role of each of the following elements of a computer system:

a. The computer hardware?

b. The programs that can be run?

c. The operating system?

🔑 5. One of the key OS abstractions that was explained using the college metaphor was Multiprogramming.

a. What is multiprogramming?

b. Give an example using the college metaphor, different from the ones mentioned in class, to explain the idea of multiprogramming. I.e. give an example of different “programs” that might be running at the same time.

🔑 6. The second definition of an operating system from class frames the OS as providing an Application Programming Interface (API). That API is the “system call interface” OS abstraction that was also explained using the college metaphor.

a. What do application programs (i.e. user programs) use the system call interface for?

b. Give an example, different from the ones mentioned in class, that uses the college metaphor to explain the system call interface.

🔑 7. Think about the key OS abstractions that were described using the College metaphor and invent a different metaphor of your own for an operating system.

a. Briefly describe your metaphor.

b. Indicate the the elements of your metaphor that will play the roles of each of the following system components:

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  | **System Component** | **Metaphor Element** |  |
|  | Hardware |  |  |
|  | User Programs |  |  |
|  | Operating System |  |  |
|  |  |  |  |

**Bootstrapping the OS:**

We now have at least a preliminary feel for what an operating system is and some of what they do. We know that the OS is a program and that we use the OS to run user/application programs. But that leaves the question of how does the OS get loaded and run? Recall, that in a stored program architecture, a program must be in main memory in order for it to be executed. So, we must get the OS into the main memory before it can be run. The bootstrap process is the series of steps by which an OS gets loaded into the main memory of the computer so that it can be executed by the fetch/decode/execute cycle.

It is not required viewing, but the video Booting an Operating System by Dhananjai Rao provides detail and perspective on the bootstrap process, in addition to what was given in class, that you might find interesting and helpful.

* <https://www.youtube.com/watch?v=7D4qiFIosWk> (11:10)

🔑 8. What is the purpose of the bootstrap process?

9. The bootstrap process relies on programs in the BIOS.

a. What does the acronym BIOS stand for?

b. Identify the three main parts of the BIOS and briefly explain the purpose of each.

c. Where is the BIOS stored? What advantage does that provide?

🏆 d. What technique allows the programs on the BIOS to run even though they are not actually in the main memory? Explain what this technique does in a sentence or two of your own words.

🔑 10. The POST, the bootstrap, the MBR bootloader and the Stage2 bootloader are all programs that play a role in the boot process for personal computers. In a few sentences of your own words, explain what each of these programs does and how they work together to get the operating system loaded and running.

🏆 11. Like all parts of computing systems, the BIOS used in the bootstrap process has evolved over time. The current state of the art for the BIOS is called the UEFI. Do a little research on UEFI and answer the following questions:

a. What does UEFI stand for?

b. In your own words, what are some of the significant enhancements that the UEFI has brought to the bootstrap process?

🏆🏆 12. You may have heard of “dual booting” your machine. When a machine is dual booted it can be booted into one of two different operating systems (e.g. windows or Linux). Usually the user is presented with a simple menu when the machine is started up that allows the choice of which OS to boot. Read a little about dual booting and discuss how it is fits within the two-stage boot loader process.

Optional: To help me improve and scope these activities for future semesters please consider providing the following feedback.

a. Approximately how much time did you spend on this activity outside of class time?

b. Please comment on any particular challenges you faced in completing this activity.