You should read Chapter 2,6,7,8 of the course text (Pillai, 2017) and Cifuentes & Bierman (2019) and then answer the questions below, adding them as evidence to your e-portfolio.

1. What factors determine whether a programming language is secure or not?
2. Could Python be classed as a secure language? Justify your answer.
3. Python would be a better language to create operating systems than C. Discuss.

1.) Secure Language Traits

I. Languages which allow direct low-level access of hardware, especially that related to memory management, for instance in C, can cause vulnerabilities such as:

- Static buffer overflows, where predefined memory sizes are breached by inputting larger variables, which may cause the overwriting of existing variables in the stack
- Heap buffer overflows, where the heap which can dynamically expand in memory begins to infringe on already used memory

This needs to be prevented through dynamic memory management within the compiler or interpreter.

II. Languages which facilitate Internet or browser use such as JavaScript, need to sanitise and hide user input so as to prevent any malicious intentions with the supplied information, in the form of an injection attack.

III. When languages provide information regarding various activities like system logs and environmental variables, it should be at an abstraction layer which prevents users of the application from easily accessing it.

2.) Python Security

I. Yes, Python is a secure language because it performs dynamic memory management, and implements various secure language techniques. However, there are still some potentially dangerous functions, such as the eval() function which executes user input without any validation.

3.) Best Language for OS Development

I. No, Python would not be a good language for developing Operating Systems because:

   - It Is an interpreted language, therefore does not store code as machine language, and hence cannot be directly run by the CPU. C is a compiled language which will is directly executable for a particular hardware architecture once a compiled file is created - .exe in Windows, .elf in Linux

II. It must be noted that in order to provide certain levels of flexibility, languages may become more vulnerable, but developers need to ensure that they use secure programming techniques