Week 4 Practice Quiz

1. Which of the following is NOT a question to ask about what you are assuming about each library function's actions?
    - What information does the library function obtain from the environment and servers?
    - What assumptions does the library function make?
    - Who wrote the library function?
    - Does the library function do what the manual says?

Correct Who wrote the module does not matter. What matters are the assumptions and the information the library relies on, and that it be used as the manual says.

2. Assuming p is a pointer variable, why is the statement "p = malloc(-128);" poor programming?
    - It is invalid and will give a compile-time error
    - This will allocate a very large amount of space.
    - You can't allocate negative amounts of space.
    - Allocating a negative amount of space frees that many bytes, so "free(p);" should be used instead

Correct The argument to malloc() is an unsigned int, not an int, so the "-128" will be interpreted as a vary large unsigned (and hence positive) number.

3. Which of the following is NOT a question about what users or remote servers will be supplying that is relevant to secure programming?
    - What certifications does the user or remote server have?
    - How can I check what the user or server is supplying for validity?
    - What am I assuming about the environment?
    - What happens if what the user or server supplies is bogus?

Correct This is irrelevant because certifications are not a guarantee that the user or server will send what the program expects. The other three all speak to the processing of what is supplied, so all are good questions to ask.

4. Which of the following should you **AVOID** whenever possible?
    - Passing pointers through a parameter list
    - Passing unsigned integers through a parameter list
    - Passing signed integers through a parameter list
    - Checking arguments passed through an interface are valid

Correct This describes something that can be checked only in the most limited way (is it the NULL pointer or not?)

5. Which of the following is NOT a language used to state specifications for formal methods?
    - C
    - Z
    - HOL
    - SPECIAL

Correct C is a programming language; the others are logic languages used to prove consistency of specifications.

6. Which principle of secure design does stepwise refinement follow?
    - Principle of fail-safe defaults, because if one module fails, the rest can compensate for the failure.
    - Principle of least common mechanism, because the modules do not share information
    - Principle of economy of mechanism, as each module performs one task
    - Principle of open design, as you can publish the details of the refinement

Correct Open design refers to releasing information about the design, and stepwise refinement is orthogonal to that question.

7. When you write a secure program, the goals must be
    - stated clearly and unambiguously
    - attainable regardless of the system on which the program is written
    - specified in a mathematical or logical language
    - written in clear English

Correct We do so to know what the program is supposed to do. The mathematical and logical languages force the first, the clarity, and the proofs and the verifiers will typically uncover most of the ambiguities. We don't have those luxuries though, so we're going to have to focus on stating the goals clearly and making sure that they don't contradict one another.