**Compare stepwise refinement to top-down and bottom-up architecture methodology.**

Stepwise refinement is a top-down process in which the problem gets broken down into smaller pieces and simplified based on a series of design decisions while expanding on the details of each succeeding level. Stepwise refinement relies upon the use of pseudocode, or structured English, to provide the details of each abstraction and aid in determining if it can continue to break down further without the implementation of code.

**Discuss how stepwise refinement links to structured design.**

Stepwise refinement is linked to structured design through the use of pseudocode. Pseudocode, or structured English, is a low-level design tool that uses natural language in a programming-language-like structure to help visualize algorithms until reaching the point where it is necessary to use an actual programming language.

**Apply stepwise refinement on the selected problem. Given a set of items, each with a weight and a value, determine the number of each item to include in a collection so that the total weight is less than or equal to a given limit, and the total value is as large as possible.**

First, create a two-dimensional matrix that represents all subsets of the items. The rows of the matrix represent the items, and the columns represent the remaining capacity. Next, loop through the matrix to calculate the possible value that can be obtained at each stage of the knapsack capacity for each combination of items. Lastly, using the completed matrix, determine the items to add to the knapsack to maximize the value carried in it.

**Rationalize on why you need to delay any language-specific activity as much as possible.**

The most significant reason for delaying the use of any programming-language-specific activity is to make the algorithm as versatile as possible so it may be used in multiple programming languages.

**References**

Dooley, J. F. (2017). [*Software development, design, and coding: With patterns, debugging, unit testing, and refactoring*](https://ashford.instructure.com/courses/94102/modules/items/4766032) (2nd ed.). Retrieved from https://www.vitalsource.com/