1. This is not a true data abstraction since the program can actually modify whatever the top element is, bypassing the abstract data type.

For instance:

Stack s = new Stack(5); // Create a new Stack object with 5 items

int i = 35; // Create an integer with value 35

pointer = \*int; // Create an integer pointer

pointer = s.top(); // Set the pointer to the top value of the Stack s

\*pointer = 45; // Set the pointer’s value to 45, which sets the Stack s’s top value to 45

1. C# properties have these advantages in comparison to accessor methods:
   1. Setters can have specific constraints, for example specifying a min/max value
   2. Getters can be defined without a setter for read only access
2. The three ways are:
   1. Qualifying the name of a library with the name of a namespace (libraryName::variableName)
   2. Qualifying individual names from a specific namespace. (using namespaceName::variableName)
   3. Qualifying all of the names available in a specific namespace (using namespace namespaceName)
3. I do not think it is a good idea, simply because you would need to explicitly define variables with the System namespace, which could be considered a waste of time.
4. A Python module isn’t really different from a Python script, it’s a file named something like “program.py” and it contains functions, variables, etc., much like a Python script would. If you did have a module named “program.py”, you would import it like this: “import program”. You could then run any function of program like this “program.doFunction()” A Java package is a bit different, as it is a compiled encapsulation of classes. A package can have many different classes inside of it. To import a package, you could do something like “import com.keaton.\*;” or “import com.keaton.brain;” where the former would import all the classes inside of the “keaton” package, and the latter would import only the “brain” class from the “keaton” package.