Don Petersen

S1 Discussion Questions

1) Explain what is meant by the syntax of a programming language. Explain what is meant by the semantics of a programming language. How are they related? (SLO 2)

According to the textbook, syntax is the rules of a language that dictate exactly how vocabulary elements of the language can be combined to form statements. An example of syntax for C# is that an integer variable is declared by using the reserved word int followed by the name of the variable and then a semicolon. (int x; -declares the integer variable x.) The semantics of a statement in a programming language define what will happen when that statement is executed. Using the example of int x; semantics define this a statement to declare the integer variable x. This is an unambiguous statement, meaning that there is only one way that this statement will ever be interpreted. Syntax and semantics are how programming languages determine how a statement should be written and how that statement is interrupted by the computer.

2) What are the primary concepts in C# that support object-oriented programming? Which of these do you find most important? Why? Can you eliminate any? (SLOs 1 & 2, EOs: 2a - 2h)

C# is an object-oriented programming language and the concepts of C# were build around the use of object oriented programming. The most important concept with an object oriented language is to understand what problem you are trying to solve and then build objects around the solutions that you come up with. Objects are built from classes, which define the object, and use attributes to store appropriate values and methods to execute statements. Other concepts of C# that support object-oriented programming include encapsulation, which protect the object from being changed by methods outside of the object, inheritance, which allows a class to create multiple classes that receive attributes and methods from their parent class but can also have their own, and polymorphism, is the idea that we can refer to multiple types of related objects over time in consistent ways. I do not believe any of these concepts can or should be eliminated, however, according to the article assigned that compares C# to C++ it mentions that C++ has the advantage of multiple inheritance.

3) What must be done in order to declare a variable? What does C# do during execution when a variable declaration is encountered? How is this similar/different from other languages you know? (SLO 2, EOs: 2a - 2h)

In order to declare a variable you follow the syntax to do so. The syntax of variable declaration is to use the reserved word for the type of variable you want to use, followed by a given name and then a semicolon. An example of a variable declaration is: int x; This example shows an integer variable named x is being declared. When a variable is encountered during execution a portion of main memory is reserved for the particular type of variable and the value of the variable is then stored. For example: in x = 8; would store the number 8 in the main memory. This is the exact same syntax as C++ but differs from VB which uses the reserved word Dim during variable declaration.

4) What do widening and narrowing mean? Which is safer and why? (SLO 2, EO 2a)

Widening and narrowing have to do with data conversion. Some data types hold more bits than others; for example, a byte is stored in 8 bits while a short is stored in 16bits. When data is converted from a type that can hold fewer bits to one that can hold more bits, the conversion is considered to be widening. Narrowing is then the opposite, conversion from larger to smaller data types. It is also then safer to make widening conversions because there is no risk of data loss. For narrowing conversions, if, for example, you tried to convert a 16 bit data type to an 8 bit data type, 8 bits would be lost in the conversion. You can’t put 16 apples in a basket that only holds 8 apples.