CS 6311 – Programming Languages I – Fall 2006
Quiz #3 (20 pts)
Monday, 9/11/6

NAME:

1. True or False. With an interpreted language making decisions at run-time, it is not possible to detect syntax errors in an interpreted language. (2 pts)

   False.

2. At what stage of compilation are errors like type errors identified? (2 pts)

   Stage 3 – Semantic analysis.

3. Which of the three language translation methods use a virtual machine? (3 pts)

   Hybrid and purely interpreted.

4. Which translation method: pure compilation or pure interpretation has slower program execution? Explain. (5 pts)

   Pure interpretation. This is because in an interpreted language each line of code must be decoded (analyzed) as it is run. This is particulary devasting for the code within a loop as the same code is decoded over and over while the program is running. However, even if the program didn't have loops pure interpretation is still slower because of the decoding process that occurs during run-time. This is in contrast to purely compiled language where the decoding to machine language is only done once before the program even begins to execute, so when it runs the code it does not need to perform the decoding step. Therefore, since the decoding for an interpreted language occurs as the program is running, an interpreted program runs 10-100 times slower than an equivalent compiled program.

   Side note: Remember an interpreter is a program that executes other programs.

5. Python uses dictionaries to create key:value pairs. What is another name for a dictionary? (2 pts)

   Associate-array, associate-memory, or hash are acceptable answers.

6. Given the following Python code that defines a class Car, what is wrong with the code that attempts to create a Car class and print out the name of the car? The comments indicate what is trying to be accomplished. Please fix the errors, if any. The errors could be anywhere in the code given. (6 pts)

```
class Car:
   "Defines a car class"

   def __init__(carName):      Change to:   def __init__(self, carname)
      name = carName # Create attribute name  Change to: self.name = carName

# end class Car

# Creating a Car object
myCar = Car("Dodge")

# accessing the name attribute of the car object
print myCar.carName            Change to: myCar.name

# The output should be "Dodge"
```