CSCE 110: Programming I

Exam #1 — List of Topics

September 24, 2011

1 Preparing for Exam #1

Below are a list of topics that Exam #1 will cover. The exam material will include all material from course lectures, course notes, labs, and quizzes. The purpose of the exam is to test your ability to read and write Python programs.

In terms of preparing for the exam, I have the following advice.

1. **Study weekly quizzes.** Make sure you understand every question on the weekly quizzes.

2. **Reading practice.** Gather all of the programs that we have discussed during class lecture. For each program that we discussed in lecture, trace through the code on paper. Next, compare the output you got on paper with the output when the program is run on the computer. If there is a match, then you understand the program. Otherwise, there is an issue with your understanding of the program that must be resolved.

3. **Writing practice.** Pick a few of the programs that we discussed during class lecture. Write a different solution for the problem. Remember, there is no one way to solve a problem.

4. **Study lab problems.** Make sure you understand the lab problems. Many of the labs required you to write several small functions in order to complete the assignment. These small functions are perfect for test questions. For example, in lab #9, part of the solution to question #1 required you to write a `is_vowel(c)` function to determine whether a character (i.e., a string of length 1) `c` is a vowel. Be able to read and write such a function for the exam.

5. **Study sample exam questions.** There will a set of sample questions to also help you study for the exam. The questions will be available on the course website by Wednesday, April 6th. We will go over a few of the questions in class on Thursday, April 7th.

If you study in the above manner, you will be well-prepared for the exam.

2 Exam #1 topics

1. Python shell
   a) Commands may be entered at the `>>>` prompt.

2. Writing your Python programs in WingWare IDE 101
a) Start a new window from the File menu by selecting New or Open.
b) Make your Python file names end explicitly with '.py'
c) To run a program from WingWare IDE 101 hit the green arrow button.

3. Errors come in three categories:
   a) Syntax errors: text that the interpreter recognizes as illegal when first reading it.
   b) Execution errors: the first illegal action is detected while running your command or program.
   c) Logical errors: the Python source code is legal, but you do not get the results you desire.

4. Python keywords (make sure you can identify them)

5. Types of data
   a) Integers: 4, 7, -7
   b) Floats: 4.0, 7.787, -10.993
   c) Strings: 'hello', "The answer is: ", [1, 2, 3], ['yes', 'no', 'maybe'], [1, 2, 'three', 4, 'five']
   d) Lists: [1, 2, 3], ['yes', 'no', 'maybe'], [1, 2, 'three', 4, 'five']
   e) Tuples: (1, 2, 3), ('yes', 'no', 5)
   f) Boolean: True, False

6. Variables: identifiers used to name Python data

7. Strings
   a) Strings are immutable. They cannot be modified.
   b) Slicing (extracting) a string:  \[\text{s}[i:j]\]:  \text{s} is a string expression and  \text{i} and  \text{j} are integer expressions representing the indexes. If  \text{i} is omitted, it defaults to 0. If  \text{j} is omitted, it defaults to the length of  \text{s}.
   c) You can also use negative indexes to retrieve a value from a string.

8. Lists and Tuples
   a) Lists are mutable, but tuples are immutable.
   b) Similarly to strings, you can slice and extract pieces of interest from list and tuples.
   c) You can also use negative indexes to retrieve values from lists and tuples.

9. Arithmetic operator precedence (from highest to lowest)
   a) +X, -X (unary identity, unary negation)
   b) ** (exponentiation)
   c) *, /, % (multiplication, division, modulo)
   d) +, - (addition, subtraction)

10. Comparison operators (<, <=, >, >=, ==, !=)
    a) These operators evaluate to either True or False.
11. Decision-making
   a) if/else
   b) if/elif/else

12. Loops/Repetition
   a) while loop
   b) for loop

13. random module:
   a) import random: allows us to use Python code to generate random numbers
   b) random.randint(a,b): generates a random integer between integers a and b.

14. + operator: works in different ways depending on the operands involved.
   a) int + int = int (integer addition)
      Example: 5 + 5 is the integer 10
   b) string + string = string (string concatenation)
      Example: 'hello' + 'goodbye' is the string 'hellogoodbye'
   c) list + list = list (list concatenation)
      Example: [1, 2, 3] + ['a', 10] is the list [1, 2, 3, 'a', 10]
   d) string + list = error
      Example: 'hello' + [1, 2, 3] is an error
   e) int + string = error
      Example: 5 + 'hello' is an error

15. Nested structures
   a) Nested loops
   b) Nested lists and tuples

16. The range function: a way to automatically generate a list of integers of interest
   a) range(1,10): returns the list [1, 2, 3, 4, 5, 6, 7, 8, 9]
   b) range(0,10,2): returns the list [0, 2, 4, 6, 8]