Quiz 1 Study Guide

Here is a list of concepts you will be expected to know. The best way to study for the exam is to (in order of priority): (a) do the practice mini-quizzes and practice quiz; (b) review the slides; and (c) review the lecture files and ensure that you understand them.

WEEK BY WEEK SUMMARY

WEEK 1: Just a High-Level Intro

No explicit questions from week 1 will be asked. Mostly background info regarding how computer programs are interpreted.

WEEK 2: Variables and Expressions

Monday

- Primitive data types: string, float, int, boolean, and None
    a. Why do data types matter?
- Variables:
    a. How do you assign values to them?
    b. How do you name them (mnemonic, snake case)?
    c. What are reserved words?
- Arithmetic operators
- Assignment operator (=)
- Operator precedence
- Common built-in functions:
    a. input() # always returns a string
    b. print() # doesn’t return anything, but outputs text to the screen
    c. int() # always returns an int (if invoked correctly)
    d. float() # always returns an int (if invoked correctly)

Know what each of the follow arithmetic operators does:

+	Addition	Adds values on either side of the operator
-	Subtraction	Subtracts right hand operand from left hand operand
*	Multiplication	Multiplies values on either side of the operator
/	Division	Divides left hand operand by right hand operand
**	Exponent	Performs exponential (power) calculation on operators
%	Modulus	Divides left hand operand by right hand operand; returns remainder

//	Quotient	Divides left hand operand by right hand operand; returns quotient

Wednesday: Lists and Tuples

Working with lists and tuples. Make sure you know how to...

- Create a list / tuple
- Access data primitives in a list / tuple
- Access data in a list of lists, tuple of tuples, tuple of lists, list of tuples
- Update, add to, and remove items from a list (append, pop, and assign to an index)
- Remember: lists and tuples have zero-based indexing!
- What does the built-in len() function do and why is it useful?

Some important list and tuple functionality listed here (Lesson 4):

- Lists
- Tuples

And hint: make sure you understand the answers to the challenge problems from lesson 4.

Friday: Introduction to Functions

Make sure that you understand...

- The difference between required (positional) and optional (keyword) arguments.
- That order matters when passing arguments into a function:
  - All positional parameters must be passed in first
  - Keyword parameters must be passed in after the positional parameters, but order doesn't matter among the keyword parameters. You also need to pass arguments in the format param_name=argument.
- That some functions require arguments and some do not
- That some functions return values and some do not
- That functions are expressions
- That functions can be nested within other functions and are evaluated from the inside out. For instance, in this expression...

  ```
  mul(add(2, mul(4, 6)), add(3, 5))
  ```

  ...the mul(4, 6) has to be evaluated before the add(2, mul(4, 6)) can be calculated — just like math. So:
  - mul(4, 6) evaluates to 24, and then
  - add(2, 24) evaluates to 26

WEEK 3: Functions

Monday & Wednesday: Creating Your Own Functions

- Why are functions useful?
- What is a function header?
- What is a function body?
- How do you **define** a function (i.e. what are the essential components of a function signature)?
- How do you **call/invoke** a function?
- What are positional parameters?
- What are keyword parameters?
- What's the difference between a parameter and an argument?
- When does a function need to return a value?
- How do you write a function from scratch?
- Questions to ask for any function:
    - a. What's its job?
    - b. Required arguments (and ordering)
    - c. Optional arguments
    - d. What does it return? (note that functions can but don't have to return values)
- What is a docstring?
- What are type hints?

Friday: Function Scope; Modules

Scope

- What is meant by "variable scope"
- Global and local variables (variable scope)

Modules

- What is a module and why is it useful?
- What do you have to do to use a module in your code?
- What are some built-in Python modules?
- Module to memorize: random(), and specifically three functions within the random module:
    - random.randint(a, b)   # picks a random integer between a and b
    - random.uniform(a, b)   # picks a random float between a and b
    - random.choice(['a', 'b', 'c', 'd', 'e'])   # picks a random element from the list