1. Operator precedence refers to the rules that determine the order in which components of an expression are evaluated. [See section 2.7 of the reading.] Some examples include:
   • >>> 4+5*3
     19
     The value 19 is returned because ‘*’ has higher precedence than ‘+’, therefore, 5*3 is evaluated first, then that result, 15, is added to 4.
   • >>> (4+5)*3
     27
     In this example, parenthesis have been used to force 4+5 to be evaluated first. Then that result, 9, is multiplied by 3.
   • >>> 3/2*4
     4
     In this example, ‘/’ (division) and ‘*’ (multiplication) have the same precedence, therefore, the expression is evaluated from left-to-right. So, first 3/2 is evaluated—and since the operands are integers, integer division is performed, returning 1. The value 1 is then multiplied by 4. You may consider these examples too:
     >>> 3/2*4.0
     4.0
     >>> 3.0/2*4
     6.0
     In both cases, the order of operations is the same as before. In 3/2*4.0, 3/2 is evaluated first using integer division, then the result is multiplied by a float (note the decimal point), returning a float. In 3.0/2*4, 3.0/2 is evaluated first, and since we have a float divided by an int, the operands are converted to floats, returning 1.5, which is then multiplied by 4 (and since one of the operands is a float, a float is returned).

[By the way, % has the same precedence as division and multiplication.]

2. The list slice operator, denoted [n:m], returns a segment of a list from the ‘n-th’ element up to, but excluding, the ‘m-th’ element. For example:
   • >>> x=[1,2,3,4,5]
     >>> x[1:4]
     [2, 3, 4]

     In this example, x is a list of length 5. The slice operator [1:4] applied to the list returns a list containing the elements of x, starting at index 1, up until index 3 (because 4 is excluded). [Remember that list indexes start at 0!]

CS125
Practice Midterm 1 Solutions
3. The correct answer is (e). To verify:
   - >>> x=['spam', 1.0, [2, 3], 4, 5.0]
     >>> x[3][1]
     Traceback (most recent call last):
       File "<stdin>", line 1, in <module>
       TypeError: 'int' object is not subscriptable

   In this example, x[3] is evaluated first, which returns the 4th element
   (because indexes start at 0) of x, which is the integer 4. The command then
   tries to use the subscript operation [1] on that value (effectively calling 4[1]),
   which is invalid.

   [Note that if you copy and paste the command from the practice quiz pdf into
   Python, you will get an invalid syntax error because the single quote is not
   rendered correctly]

4. You can check your answer by using the Python Visualizer. [Here I changed 'hide
   frames of exited functions' to 'show frames of exited functions']. You should step
   through the code and verify all steps of your solution, including noticing where
   something should be crossed out (the visualizer doesn't keep old values around, but
   on paper nothing should be erased.)
5. Here is a function that satisfies the description and some code to demonstrate it:

```python
def avgSum(mylist):
    mysum = float(sum(mylist))
    print mysum
    myave = mysum / len(mylist)
    return myave
```

```python
x=[1,1,2]
ave=avgSum(x)
print 'ave is', ave
```

The output of the program is:

```
4.0
ave is 1.3333333333
```

Note that the function converts the sum to a float, ensuring that float division is performed when computing the average (without converting to float, the function would have returned the int 1, the result of performing the integer division 4/3).

6. Here is one possible solution:

```python
def mystery():
    x=4
    y=5
    x=10
    mystery()
```