1. Vocabulary Matching: (15 points)

Write the number from the correct definition in the blank next to each term on the left:

<table>
<thead>
<tr>
<th>15</th>
<th>compound data type</th>
<th>1</th>
<th>slice</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>traverse</td>
<td>3</td>
<td>mutable</td>
</tr>
<tr>
<td>10</td>
<td>increment</td>
<td>11</td>
<td>decrement</td>
</tr>
<tr>
<td>12</td>
<td>element</td>
<td>6</td>
<td>aliases</td>
</tr>
<tr>
<td>8</td>
<td>sequence</td>
<td>9</td>
<td>nested list</td>
</tr>
<tr>
<td>14</td>
<td>clone</td>
<td>13</td>
<td>immutable type</td>
</tr>
<tr>
<td>2</td>
<td>dictionary</td>
<td>4</td>
<td>recursion</td>
</tr>
<tr>
<td>5</td>
<td>iteration</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1. A part of a string specified by a range of indices.
2. A collection of key-value pairs that maps from keys to values.
3. A compound data type whose elements can be assigned new values.
4. The process of calling the function that is currently executing.
5. Repeated execution of a set of statements using either a recursive function call or a loop.
6. Multiple variables that contain references to the same object.
7. To iterate through the elements of a set, performing a similar operation on each.
8. Any of the data types that consist of an ordered set of elements, with each element identified by an index.
9. A list that is an element of another list.
10. To increase the value of a variable by one.
11. To decrease the value of a variable by one.
12. One of the values in a list (or other sequence). The bracket operator selects an ________ of a list.
13. A type in which the elements cannot be modified. Assignments to elements or slices of these types cause an error.
14. To create a new object that has the same value as an existing object.
15. A data type in which the values are made up of components, or elements, that are themselves values.
2. Write Code (15 points)
Write a function **return_smallest** that accepts 3 parameters \((x,y,z)\) and returns the smallest of the three. For example, **return_smallest**\((7, -34, 23.8)\) should return -34. Make sure that your function works for test cases such as **return_smallest**\((5,5,4)\).

```python
def return_smallest(x,y,z):
    if(x <= y and x <= z):
        return(x)
    if(y <= x and y <= z):
        return(y)
    if(z <= y and z <= x):
        return(z)
```

**grading:** 5 points – Correct def statement, has one or more return statements.
5 points – works correctly for differing values e.g. \((5,6,7)\) or \((-1,10,11)\)
5 points – works correctly for same values, eg: \((5,5,7)\) and \((5,5,5)\)
-1 point for minor syntax errors.

3. Program Comprehension (3 points)
```python
def n_lines(n):
    print "Line!"
    if n >= 0:
        n_lines(n-1)
```

How many times will the string “Line!” be printed when *n_lines* is called with *n*\(=4\)?
Number______\(6\)______

4. Write Code (2 points)
Write a function with infinite recursion named **run_forever**. Your function should have no parameters, and it should run forever when called (on an ideal computer, in a real computer it would eventually run out of memory.) You may add a print statement if you wish.
```python
def run_forever():
    run_forever()
5. Robot Directions (10 points)
The following code makes the robot drive the trajectory drawn in the box to the right.

```python
def turn90degrees():
    turnRight(1, 1)

def nudge(x):
    forward(1, x)

nudge(1)
turn90degrees()
nudge(1)
nudge(2)
```

Draw the robot's trajectory when the following code is executed. Start the robot in the middle of the box and use arrow heads (as above) to indicate each movement.

```python
def turn90degrees():
    turnRight(1, 1)

def nudge(x):
    forward(1, x)

nums = [1,2,3,4]
for I in nums:
    if (I % 2 == 0):
        turn90degrees()
    nudge(I)
```

Grading: -2 for each incorrect turn, -1 point for each incorrect line segment.

6. Fill in the blank (2 points)
In python, the = operator performs ____Assignment________ while the == operator performs __Comparison / Equality Check________________.
7. Python Expression Evaluation (20 points)
For this question, assume the following statements have already been entered and interpreted:
\[ a = [ True, 7, ["Cherry", "Apple", "Plum"], 56, [4, 5, 6], 84 ] \]
\[ b = a \]
\[ c = a[0:4] \]
\[ d = a[2] \]
\[ d[2] = "Peach" \]

Pretend that you are the Python Interpreter (IDLE window). What do you print or return when each of the following statements are entered?

Example:  \[ a[0] \]
\[ \text{Result: } \boxed{True} \]

Example:  \[ a[4:6] \]
\[ \text{Result: } \boxed{[4,5,6,84]} \]

1. \[ a[4][0] \]
\[ \text{Result: } \boxed{4} \]

2. \[ d \]
\[ \text{Result: } \boxed{["Cherry", "Apple", "Peach"]} \]

3. \[ c \]
\[ \text{Result: } \boxed{[True, 7, ["Cherry", "Apple", "Peach"], 56]} \]

4. \[ a[2][2] \]
\[ \text{Result: } \boxed{"Peach" \text{ or } 'Peach'} \]

5. \[ b[:2] \]
\[ \text{Result: } \boxed{[True, 7]} \]

6. \[ b[-2] \]
\[ \text{Result: } \boxed{[4,5,6]} \]

7. \[ c[-2] \]
\[ \text{Result: } \boxed{["Cherry", "Apple", "Peach"]} \]

8. \[ \text{print } "Pumpkin %.3f" \text{ %3.1459} \]
\[ \text{Result: } \boxed{\text{Pumpkin 3.146}} \]

9. \[ (5 > 10) \text{ or } (5 > 3) \]
\[ \text{Result: } \boxed{True} \]

10. \[ 34 \% 10 \]
\[ \text{Result: } \boxed{4} \]

Grading: -2 points if wrong. -1 point if missing/adding quotes, or getting 3.145 on question 8. If they mess up the “Peach”/”Plum” bit, take off 2 points the first time, and only 1 point the subsequent times.
8. Write Code (10 points)

Write a function `changeLetter(aString, index, newLetter)` that will replace the letter stored at index in `aString` with the contents of `newLetter` and return the new string 

*without modifying the original string!* For example, `changeLetter("Python is great!", 10, "G")` will return the string "Python is Great!"

def changeLetter(aString, index, newLetter):
    return( aString[:index] + newLetter + aString[index+1:] )

grazing: 5 points for having a correct def line w/ parameters, and a return statement. -1 point for missing the colon.

5 points for working (even if off by one). -2 point for being off by a letter.

9. Write Code (5 points)

Write a function `changeValue(aList, index, newValue)` that will replace the element stored at index in `aList` with the contents of `newValue`. It should NOT return the list. For example after the following commands:

```python
a = [5, True,"Test",10]
changeValue( a, 3, "Hi!")
```

The list `a` will be [5, True,"Test","Hi!"]

def changeValue(aList, index, newValue):
    aList[index] = newValue

grazing: 2 points for correct def line, -1 for missing the colon.

3 points for working. -2 for returning anything other than a blank or None.
10. Write Code! (20 points)

Write a function called `roboFlute` that takes no parameters. The `roboFlute` function will watch each of the robot's 3 light sensors (values obtained using the `getLight("loc")` function and specifying a location out of the set ("left" / "right" / "center") ) and play a beep that is $\frac{1}{2}$ second long if a light sensor is covered. You know a light sensor is covered when the value it returns is larger than 1000. Each of the three sensors should play a different note, as follows: “left” = 800Hz, “center” = 440Hz, “right” = 220Hz.

The `robotFlute` function should perform the above actions for 25 seconds and then return.

API Hints: `beep(time_in_seconds, frequency_in_Hz), \quad value = getLight("location")`

def roboFlute():
    while( timeRemaining(25) ):
        if (getLight("left") > 1000 ):
            beep(0.5, 800)
        if (getLight("center") > 1000 ):
            beep(0.5, 440)
        if (getLight("right") > 1000 ):
            beep(0.5, 220)

1 free point if they attempted to answer.

1 points for getting the def line right and not returning anything.

6 points for running for 25 seconds.

4 points (12 total) for each sensor triggering the appropriate beep.
11. Write Code! (15 points)

Write a function `reverseList(aList)` that will return a reversed copy of `aList`. For example, after the following:

\[ a = [5, 10, True, "Hi!"] \]

\[ b = reverseList(a) \]

The list \[b = ["Hi!", True, 10, 5]\], while \[a = [5, 10, True, "Hi!"\].

Simple way:

```python
def reverseList(aList):
    return (aList[len(aList):0:-1])  # or return( aList[::-1] )
```

Another way:

```python
def reverseList(aList):
    bList = []
    for x in aList:
        bList = [x] + bList
    return(bList)
```

Grading: 7 points for correct def line w/ parameters and a return statement.

8 points for working.
12. Write Code: (10 Points)

Write a function `findJ(aString)` that uses a `while` loop to find the index of the first occurrence of the letter 'J' in the `aString` parameter. The function should return the index it found. For example, `findJ("This is Jays String")` should return the number 8. Note that you should find both UPPERCASE J's and lowercase j's! If you do not find a J or a j you should return -1.

```python
def findJ(aString):
    index = 0
    while(index < len(aString)):
        if (aString[index] == 'j' or aString[index] == 'J'):
            return(index)
        index = index + 1
    return(-1)
```

Grading: 3 points for def, parameter, and return.
3 points for the while loop and index counter.
4 points for working correctly.

Extra Credit (1 point each)

What is the one sensor on the Scribbler that does not detect light of one form or another?
`getStall()` (or `getBattery()` although technically the voltage sensor is on the fluke.)

What is the decimal representation of the binary number `{ 101101}”? 45

What is the hexadecimal representation of the decimal number 34? 22

What does CSS stand for? Cascading Style Sheets