15 Points – Do any 15; do more for extra credit. In this problem assume that all of the statements \( S = \text{"GREEN"}, \ L = [3, (6, 7.2), \text{"FROG"}, 8, \text{"TOAD"}], \) and \( T = (9.5, [2, 5, 7], \text{"NEWT"}) \) have been executed. Show the computed result after each statement is executed (all are independent of one another). Indicate where the computation fails because of some form of error. Be careful about the type of the result, particularly integers and floats, and put proper quotes around string results.

<table>
<thead>
<tr>
<th>Question</th>
<th>Answer</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. ( N = T[0] \times 2 )</td>
<td>( 19.0 ) (must be float)</td>
</tr>
<tr>
<td>2. ( N = L[1][1] + 1 )</td>
<td>( 8.2 ) (must be float)</td>
</tr>
<tr>
<td>3. ( N = L[0] / 2 )</td>
<td>1 (must be int)</td>
</tr>
<tr>
<td>4. ( N = \text{len}(L) )</td>
<td>5 (L is: [int, tuple, string, int, string])</td>
</tr>
<tr>
<td>5. ( N = L[-1] )</td>
<td>&quot;TOAD&quot; (last item in list)</td>
</tr>
<tr>
<td>6. ( N = \text{len}(T[1]) )</td>
<td>3 (length of [2,5,7])</td>
</tr>
<tr>
<td>7. ( N = T[\text{len}(T)-1] )</td>
<td>&quot;NEWT&quot; (last item in tuple)</td>
</tr>
<tr>
<td>8. ( N = L[6] )</td>
<td>error (off end of list)</td>
</tr>
<tr>
<td>9. ( N = L[4][3] + \text{&quot;O&quot;} + L[2][3] )</td>
<td>&quot;DOG&quot;</td>
</tr>
<tr>
<td>10. ( N = \text{len}(T[0]) )</td>
<td>error (can’t take length of float)</td>
</tr>
<tr>
<td>11. ( N = \text{len}(S) )</td>
<td>5</td>
</tr>
<tr>
<td>12. ( N = S + \text{&quot; &quot;} + L[2] )</td>
<td>&quot;GREEN FROG&quot;</td>
</tr>
<tr>
<td>13. ( S = S + \text{&quot;!&quot;} )</td>
<td>&quot;GREEN!&quot;</td>
</tr>
<tr>
<td>14. ( T = T + (5, 6) )</td>
<td>((9.5, [2,5,7], \text{&quot;NEWT&quot;}, 5, 6))</td>
</tr>
<tr>
<td>15. ( L[0] = L[0] + 1 )</td>
<td>([4, (6, 7.2), \text{&quot;FROG&quot;}, 8, \text{&quot;TOAD&quot;}])</td>
</tr>
<tr>
<td>16. ( T[0] = T[0] + 1 )</td>
<td>error (tuples are immutable)</td>
</tr>
<tr>
<td>17. ( S[0] = \text{&quot;P&quot;} )</td>
<td>error (strings are immutable)</td>
</tr>
<tr>
<td>18. ( N = [X+1 \text{ for } X \text{ in range}(5)] )</td>
<td>([1,2,3,4,5])</td>
</tr>
<tr>
<td>19. ( N = [Z \text{ for } Z \text{ in } \text{&quot;GOAT&quot;}] )</td>
<td>([\text{&quot;G&quot;}, \text{&quot;O&quot;}, \text{&quot;A&quot;}, \text{&quot;T&quot;}])</td>
</tr>
<tr>
<td>20. ( N = [0 \text{ for } X \text{ in range}(4)] )</td>
<td>([0,0,0,0])</td>
</tr>
</tbody>
</table>
5 points – What is printed out by the following code fragment, assuming that variable $S1$ currently equals 4 and variable $S2$ currently equals 7?

```python
if (S1 == S2): print "Equal"
if (S1 < S2): print "Less"
if (S1 > S2): print "Greater"
```

Answer:

```python
if (S1 < S2): print "Less"
if (S1 > S2): print "Greater"
```

15 points – Change the code fragment to make it run more efficiently. That is, mark up the code above to make it run faster/better (do NOT rewrite the code fragment). Make no assumptions about the contents of variables $S1$ and $S2$.

Answer: Replace second if with elif, replace if (S1 > S2) with else.

Final code (after marking up earlier section):

```python
if (S1 == S2): print "Equal"
elif (S1 < S2): print "Less"
else: print "Greater"
```

20 Points – Completely rewrite the following code fragment to perform exactly the same task using a list comprehension instead of a while-loop (variable Result should end up containing exactly the same values as those generated in the code below):

```python
Result = []
P = 3
while (P < 100):
    Result = Result + [P]
P = P + 5
```

```python
Result = [P for P in range(3,100,5)]
```

10 Points – Show what is printed out as the result from calling Main()

```python
def Frog(X,Y,Z):
    print X+Y*Z
    return

def Toad(Z,Y,X):
    Frog(Y,X,Z)
    return

def Main():
    Frog(5,2,3)
    Toad(5,6,7)
```
10 Points – Which of the following expressions generate a random integer from 4 through 12, inclusive? (Mark “yes” or “no” next to each line.)

A. `range(4, 12)`  
   NO

B. `random.randrange(4, 12)`  
   NO

C. `random.randrange(4, 13)`  
   YES

D. `random.random(4, 12)`  
   NO

E. `int(random.random() * 9) + 4`  
   YES

F. `random.number(4, 12)`  
   NO

G. `random.choice(range(4, 13))`  
   YES

H. `random.choice([4, 5, 6, 7, 8, 9, 10, 11, 12])`  
   YES

I. `random.random() * 12 + 4`  
   NO

J. `-random.randrange(-12, -3)`  
   YES

25 Points – This function scans every pixel in a canvas. In the indicated area, write new code to first increase green by 20 if red+blue is brighter than green, and then separately increase red by 40 if blue is brighter than red but otherwise decrease red by 30.

```python
def ScanAllPixels (Canvas):
    for Y in range(getHeight(Canvas)):
        for X in range(getWidth(Canvas)):
            PX = getPixel(Canvas, X, Y)
            R = getRed(PX)
            G = getGreen(PX)
            B = getBlue(PX)

            if (R+B > G):
                G = G + 20

            if (B > R):
                R = R + 40
            else:
                R = R - 30

            setRed(PX, R)
            setGreen(PX, G)
            setBlue(PX, B)
    repaint(Canvas)
    return
```