15 Points – Answer any 15 questions. Answer more for extra credit. For each statement show the computed result (½ point) and the data type (½ point) of the result (int, float, bool, list, string, tuple, etc.) Questions are all independent of one another. If a calculation would result in an error, answer ERROR in the Result box and leave the Type box empty. Variables have values as follows:

\[
\begin{align*}
\text{DOG} &= 7 \\
\text{CAT} &= 5.0 \\
\text{BAT} &= [6, \text{DOG}, "\text{CAT}", 3.4, 5] \\
\text{ANT} &= (5,1,8,9,3) \\
\text{COW} &= \{5:12.4, "\text{FLY}":"\text{BUZZ}" , 7:8, 3:"\text{AARDVARK}"\} \\
\text{FOX} &= "\text{GIRAFFE}" \\
\end{align*}
\]

<table>
<thead>
<tr>
<th>Result</th>
<th>Type</th>
<th>Statement</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>int</td>
<td>DOG + 1</td>
</tr>
<tr>
<td>8.0</td>
<td>float</td>
<td>DOG + 1.0</td>
</tr>
<tr>
<td>3.5</td>
<td>float</td>
<td>DOG / 2</td>
</tr>
<tr>
<td>3</td>
<td>int</td>
<td>DOG // 2</td>
</tr>
<tr>
<td>3.5</td>
<td>float</td>
<td>DOG / 2.0</td>
</tr>
<tr>
<td>3.0</td>
<td>float</td>
<td>DOG // 2.0</td>
</tr>
<tr>
<td>12.0</td>
<td>float</td>
<td>DOG + CAT</td>
</tr>
<tr>
<td>False</td>
<td>bool</td>
<td>DOG &lt; CAT</td>
</tr>
<tr>
<td>5</td>
<td>int</td>
<td>BAT[-1]</td>
</tr>
<tr>
<td>ERROR</td>
<td></td>
<td>BAT[6]</td>
</tr>
<tr>
<td>3.4</td>
<td>float</td>
<td>BAT[ANT[4]]</td>
</tr>
<tr>
<td>12.4</td>
<td>float</td>
<td>COW[5]</td>
</tr>
<tr>
<td>ERROR</td>
<td></td>
<td>COW[8]</td>
</tr>
<tr>
<td>7</td>
<td>int</td>
<td>len(FOX)</td>
</tr>
<tr>
<td>5</td>
<td>int</td>
<td>len(ANT)</td>
</tr>
<tr>
<td>ERROR</td>
<td></td>
<td>len(DOG)</td>
</tr>
<tr>
<td>&quot;GIRA&quot;</td>
<td>string</td>
<td>FOX[:4]</td>
</tr>
<tr>
<td>&quot;AFFE&quot;</td>
<td>string</td>
<td>FOX[3:]</td>
</tr>
<tr>
<td>[0,1,2,3,4]</td>
<td>list</td>
<td>list(range(BAT[4]))</td>
</tr>
<tr>
<td>[0,7,14]</td>
<td>list</td>
<td>list(range(0,20,DOG))</td>
</tr>
<tr>
<td>[]</td>
<td>list</td>
<td>list(range(DOG,ANT[0]))</td>
</tr>
<tr>
<td>[0,0,0,0,0,0,0]</td>
<td>list</td>
<td>[0] * DOG</td>
</tr>
<tr>
<td>[&quot;B&quot;,&quot;U&quot;,&quot;Z&quot;,&quot;Z&quot;]</td>
<td>list</td>
<td>[X for X in COW[&quot;FLY&quot;]]</td>
</tr>
<tr>
<td>[7,8,9]</td>
<td>list</td>
<td>[DOG + X for X in range(3)]</td>
</tr>
<tr>
<td>[0,1,2,3,4]</td>
<td>list</td>
<td>[Q for Q in range(len(ANT))]</td>
</tr>
</tbody>
</table>

-½ point for omitting .0 on whole floats, quotes on strings, or brackets on lists.
15 Points – Consider the following program:

```python
import math

def F(N):
    print (N)
    print (math.sqrt(N))
    return

def Main():
    print (F(input("Enter a number --- ")))
    return
```

When the program is run, and the user types 5 at the input, the following error message occurs:

```python
>>> Main()
Enter a number --- 5
5
Traceback (most recent call last):
  File "<pyshell#2>", line 1, in <module>
    Main()
  File "C:/Users/Verts/Desktop/Tests/P2.py", line 9, in Main
    print (F(input("Enter a number --- ")))
  File "C:/Users/Verts/Desktop/Tests/P2.py", line 5, in F
    print (math.sqrt(N))
TypeError: must be real number, not str
```

A. (5 points) **What** is the error that Python is identifying as the problem?

   Python is objecting to `math.sqrt` being given a **string** instead of a **number**.

B. (5 points) **Where** is the error actually located?

   The error is in the `print` statement in `Main`, where the string result from the `input` function is not converted to a number before being passed to function `F`.

C. (5 points) **How** do you fix the error?

   Convert the result of `input` to a number:

   ```python
   Either:
   print (F(int(input("Enter a number --- "))))
   or:
   print (F(float(input("Enter a number --- "))))
   ```
<3>  5 Points – Rewrite the following function as a lambda expression:

```python
def Frog (X):
    return 2 * X + 1

Frog = lambda X : 2 * X + 1
```

-1 point per error, including forgetting to assign result to Frog, using different variable names, syntax errors, etc.

<4>  5 Points – Rewrite the code fragment below as a list comprehension:

```python
Q = []
for Z in range(2,12,5):
    Q = Q + [Z*Z + 5]

Q = [Z*Z + 5 for Z in range(2,12,5)]
```

-1 point per error, including forgetting to assign result to Q, using different variable names, syntax errors, etc.

<5>  5 Points – Rewrite the code fragment below as a for-loop:

```python
Result = 0
Index = 0
while (Index < 100):
    Result = Result + math.sqrt(Index)
    Index = Index + 3

Result = 0
for Index in range(0,100,3):
    Result = Result + [math.sqrt(Index)]
either
    Result.append(math.sqrt(Index))
or
```

-1 point per error, including getting the range arguments wrong, syntax, etc.

<6>  5 Points – What is printed by the following code when Main is run?

```python
def Main():
    Nothing!
    X = 45
    Y = 92
    if (X > Y):
        print (X+Y)
    else:
        X = 102
    return
```

Nothing!
10 Points – In the Sierpinski project, you had a choice of representing 2D points either by using separate variables for $x$ and $y$ or by using a list of two numbers where the first item in the list is $x$ and the second is $y$.

A. Write the code to assign to $P$ the average of points $P$ and $Q$ under the **first method** (that is, $P$ is represented by distinct variables $PX$ and $PY$ and $Q$ by distinct variables $QX$ and $QY$).

$$PX = (PX + QX) \div 2$$  
# Average the X
$$PY = (PY + QY) \div 2$$  
# Average the Y

3 points for doing the Xs, 2 points for doing the Ys. Float division (/) is allowed in place of integer division (\div). Within each group, -1 point per error (wrong variable names, omitting parentheses, etc.). Trailing comments not required.

B. Write the code to assign to $P$ the average of points $P$ and $Q$ under the **second method** (that is, $P$ has the value of the list $[px, py]$ and $Q$ the value of the list $[qx, qy]$).

$$P[0] = (P[0] + Q[0]) \div 2$$  
# Average the X
$$P[1] = (P[1] + Q[1]) \div 2$$  
# Average the Y

3 points for doing the Xs, 2 points for doing the Ys. Float division (/) is allowed in place of integer division (\div). Within each group, -1 point per error (wrong variable names, omitting parentheses, etc.). Trailing comments not required.

5 Points – The two code fragments below are almost but not quite the same. What is printed out by each one?

```
I = 0
while (I <= 5):
    print (I)
    I = I + 1
```

```
I = 1
while (I < 5):
    print (I)
    I = I + 1
```

**3 points**
0
1
2
3
4
5

**2 points**
1
2
3
4

15 Points – Assume that you have successfully created and debugged the Graphics.py library that you used in the later lab assignments, containing functions setPixel, addLine, HorizontalLine, etc. Complete the function below to paint a solid rectangle where X1,Y1 are the coordinates of the upper-left corner and X2,Y2 are the coordinates of the lower-right corner, with interior color InColor and a 1-pixel border of ExColor, as shown in the following image (X1,Y1 is <8,4> and X2,Y2 is <24,13> with InColor a light gray and ExColor a dark gray):
def addBox (Canvas,X1,Y1,X2,Y2,InColor=white,ExColor=black):

    # Any of the following solutions may be used for the filled 
    # rectangle. Students are allowed to fill the entire box, 
    # or they may try to fill only the interior of the box. 
    # There may be other legal variations. Accept any legal 
    # variation. 
    # 
    # 7 points: -1 point for each error, depending on which 
    # implementation is chosen, including syntax, getting the 
    # coordinates wrong, etc.

    for Y in range(Y1,Y2+1):       # Fill entire box 
        HorizontalLine(Canvas,X1,X2,Y,InColor)

    or

    for Y in range(Y1+1,Y2):       # Fill interior only 
        HorizontalLine(Canvas,X1+1,X2-1,Y,InColor)

    or

    for Y in range(Y1,Y2+1):       
        addLine(Canvas,X1,Y,X2,Y,InColor)

    or

    for X in range(X1,X2+1):
        VerticalLine(Canvas,X,Y1,Y2,InColor)

    or

    for Y in range(Y1,Y2+1):
        for X in range(X1,X2+1):
            setPixel(Canvas,X,Y,InColor)

    etc.

    # Any of the following solutions may be used for the 
    # outline. Accept any legal variation.
    #
    # 8 points, 2 points per line: -1 point for each error, 
    # depending on which implementation is chosen, including 
    # syntax, getting the coordinates wrong, etc.

    addLine(Canvas,X1,Y1,X1,Y2,ExColor)        # Top line 
    addLine(Canvas,X1,Y2,X2,Y2,ExColor)        # Bottom line 
    addLine(Canvas,X1,X1,X2,Y2,ExColor)        # Left line 
    addLine(Canvas,X2,Y1,X2,Y2,ExColor)        # Right line 

    or

    HorizontalLine(Canvas,X1,X2,Y1,ExColor)    # Top line 
    HorizontalLine(Canvas,X1,X2,Y2,ExColor)    # Bottom line 
    VerticalLine(Canvas,X1,Y1,X2,ExColor)      # Left line 
    VerticalLine(Canvas,X2,Y1,Y2,ExColor)      # Right line 

    return
20 Points – 2 points each answer. What is printed by the following program when `Main` is called? There will be exactly 10 lines printed; write the first thing printed in the slot for #1, the second thing printed in the slot for #2, etc.

```python
frog = 0
def F1 (M,N=2):
    global Frog
    Q = M + Frog
    Frog = Q + N
    print (Frog)
    return
    # Your Answers:
    # 1: 8

def F2 (P,Q="Toad"):
    global Frog
    if (Q == "Newt") or (Q == "Frog"):
        F1 (Frog)
    else:
        Frog = P - 5
    print (P)
    return
    # 2: 14
    # 3: 3
    # 4: -2
    # 5: 3

def Main():
    F1 (3,5)
    F1 (4)
    F2 (3)
    F2 (3,"Newt")
    F1 (2,1)
    F2 (3,"Frog")
    F1 (1)
    F1 (1)
    return
    # 6: 1
    # 7: 4
    # 8: 3
    # 9: 7
    # 10: 10
```