20 Points – Do any 20; do more for extra credit. Correct answers are worth +1 point, blank answers are worth 0 points, but wrong answers are incur a −½ point penalty; if you don’t know an answer, leaving it blank is usually better than a bad guess. The following statements have all been executed:

Orange = 15
Lemon = 6.5
Lime = "Avocado and Salsa"
Banana = [9, "Kiwi", 7.5, "Grape", [5, "9", 3], -12]
Apple = ([1,2,3],[4,5,6],[7,8,9])

Show the computed result for each problem; all are independent of one another. Indicate where a computation fails because of some form of error. Be careful about the type of the result, particularly int, float, long, bool, and complex types, and put proper quotes (either ‘ or ”) around string results, square brackets around lists, and parentheses around tuples.

<table>
<thead>
<tr>
<th>Question</th>
<th>Answer</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Orange + Lemon</td>
<td>21.5</td>
</tr>
<tr>
<td>2. Orange * 2</td>
<td>30</td>
</tr>
<tr>
<td>3. Orange * 2.0</td>
<td>30.0</td>
</tr>
<tr>
<td>4. Orange / 2</td>
<td>7.5</td>
</tr>
<tr>
<td>5. Orange // 2</td>
<td>7</td>
</tr>
<tr>
<td>6. Orange // 2.0</td>
<td>7.0</td>
</tr>
<tr>
<td>7. Orange + Lime</td>
<td>ERROR</td>
</tr>
<tr>
<td>8. len(Lemon)</td>
<td>ERROR</td>
</tr>
<tr>
<td>9. len(Lime)</td>
<td>17</td>
</tr>
<tr>
<td>10. len(Banana)</td>
<td>6</td>
</tr>
<tr>
<td>11. len(Apple)</td>
<td>3</td>
</tr>
<tr>
<td>12. Lime[1]</td>
<td>&quot;v&quot;</td>
</tr>
<tr>
<td>14. Apple[1]</td>
<td>[4,5,6]</td>
</tr>
<tr>
<td>15. Banana[1][1]</td>
<td>&quot;i&quot;</td>
</tr>
<tr>
<td>16. len(Banana[3])</td>
<td>5</td>
</tr>
<tr>
<td>17. Banana[-1]</td>
<td>-12</td>
</tr>
<tr>
<td>21. Banana + Apple</td>
<td>ERROR</td>
</tr>
<tr>
<td>22. Apple[0] + Apple[1]</td>
<td>[1,2,3,4,5,6]</td>
</tr>
<tr>
<td>23. range(Banana[0])</td>
<td>[0,1,2,3,4,5,6,7,8]</td>
</tr>
<tr>
<td>24. range(Banana[-1],Orange,4)</td>
<td>[-12,-8,-4,0,4,8,12]</td>
</tr>
<tr>
<td>25. range(Orange,3)</td>
<td>[]</td>
</tr>
</tbody>
</table>
<2> 15 Points – Convert the while-loop below into a for-loop that does the same thing. Your solution needs at most two lines of Python code.

```
Count = 8
while (Count < 96):
    print (Count, math.sqrt(Count))
    Count = Count + 7
```

- or -

```
for Count in range(8,96,7):
    print (Count, math.sqrt(Count))
```

Remove points as follows:
-3 For including explicit `Count = 8` statement before for-loop.
-3 For including explicit `Count = Count + 7` statement inside for-loop.
-2 For omitting range function (i.e., `for Count in (8,96,7):`)
-1 Per syntax error (omitting : at end of for-loop, omitting parentheses on print, capitalization errors, etc.

Do not penalize for including `import math` at beginning (irrelevant code).

<3> 20 Points – Show what is printed out as the result from calling `Main()` (four lines total): 5 points per answer.

```
def F1(A,B,C=5):
    D = A + B * C
    return D + 1

def F2(B,D=1,A=4):
    return F1(D,A,B)

def F3(C,D,A):
    B = F1(C,A, F2(D))
    return B

def Main():
    print (F1(5,2,3))  # Answer #1: 12
    print (F2(4,8))    # Answer #2: 25
    print (F3(3,2,4))  # Answer #3: 44
    print (F1(6,-1))   # Answer #4: 2
```

– Page 2 –
15 Points – Trace the following flowchart. Any time a variable value is updated, write the new value in the appropriate box and crossing out any previous value. There will be four lines printed as a result of running the program; show the values of the variables in each of those lines. 1 point per output value, 1 point per variable box.
<5> 15 Points – Convert the flowchart on the previous page into the equivalent and correct Python 3 code. Your solution won’t need any function definitions.

```python
T = 0
P = 1
N = 0

while N < 4:
    N = N + 1
    P = P * 2
    T = T + P

print (N,P,T)
```

Remove 1 point for each syntax error (indentation, omitting colons or parentheses, using the wrong symbol for multiplication, capitalization, etc.), but do not go below zero.
15 Points – Complete the following function to return **True** if \( N \) is greater than or equal to 0 and less than or equal to 100, but it must return **False** if \( N \) is either less than zero or greater than 100. Your solution must **NOT** contain any **print** statements.

Any of the following solutions are acceptable. There may be other valid approaches as well. Once a solution type has been identified, remove 1 point per error (using a **print**, omitting colons on **if**, **elif**, or **else** statements, switching <, <=, >, >= as appropriate, omitting parentheses on clauses using **and**, capitalization errors on **True** or **False**, etc.)

```python
def MyFunction(N):
    if N < 0:
        Result = False
    else:
        if N > 100:
            Result = False
        else:
            Result = True
    return Result
```

```python
def MyFunction(N):
    if N < 0: return False
    elif N > 100: return False
    else: return True
```

```python
def MyFunction(N):
    if (N >= 0) and (N <= 100):
        Result = True
    else:
        Result = False
    return Result
```

```python
def MyFunction(N):
    if N < 0: return False
    if N > 100: return False
    return True
```

```python
def MyFunction(N):
    return (N >= 0) and (N <= 100)
```