Question 1.  [7 marks]

The following python statements have been executed:

tweet1 = 'Computer science students organize #UofT Hacks to create new solutions'
tweet2 = 'Meet Ross, the @IBMWatson-powered lawyer @IBM'

# docstrings removed to save space
def f(x, y, z):
    return x * y * z + 3

def g(x):
    return x % 2

x = 8

Beside each code fragment in the table below, write what is printed when the code fragment is executed. If the code would cause an error, write ERROR and give a brief explanation.

<table>
<thead>
<tr>
<th>Code</th>
<th>Output or Cause of Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>print('problem' not in tweet1 and 'lawyer' in tweet2)</td>
<td>True</td>
</tr>
<tr>
<td>print(5 // 3 * 6 == 10)</td>
<td>False</td>
</tr>
<tr>
<td>print(tweet1[3].islower() and tweet2[-len(tweet2)].islower())</td>
<td>False</td>
</tr>
<tr>
<td>print(tweet1[3].islower() or tweet2[len(tweet2)].islower())</td>
<td>True</td>
</tr>
<tr>
<td>print(tweet2[len(tweet2)].islower() or tweet1[3].islower())</td>
<td>ERROR: index out of range</td>
</tr>
<tr>
<td>print(g(x))</td>
<td>0</td>
</tr>
<tr>
<td>print(f(x * g(x), 1, x))</td>
<td>3</td>
</tr>
</tbody>
</table>
Question 2.  [3 marks]

Assume that variable \( n \) refers to an \texttt{int} and that \( n \geq 0 \). Consider the following python code:

```python
total = 0
i = 0
while i <= n:
    total = total + i
    i = i + 2
```

Rewrite this code so that it uses a \texttt{for} loop instead of a \texttt{while} loop.

```python
total = 0
for i in range(0, n+1, 2):  # alternate: for i in range(0, n+2, 2):
    total = total + i
```
Question 3.  [8 marks]

In the Republic of Danmania, the cost of mailing a letter is 40 coins for letters weighing up to 30 g (inclusive) and 55 coins for letters over 30 g and up to 50 g (inclusive). Letters weighing more than 50 g cost 55 coins plus an additional 25 coins for each extra 50 g or part thereof. For example, a 100 g letter would cost 80 coins to mail (55 coins for the first 50 g plus an additional 25 coins for the next 50 g). A 101 g letter would cost 105 coins to mail (55 coins for the first 50 g plus an additional 25 coins for the next 50 g and then an additional 25 coins for the extra 1 g, which is part of the next 50 g increment).

Using the Function Design Recipe that you have learned in the course, write a complete python function named coins_required that returns the number of coins required to mail a letter with a given integer weight in grams (g).

Your function may assume that when it is called, it is given a positive integer letter weight. You may assume that the math module has been imported.

def coins_required(weight):
    """ (int) -> int

    Return the postage required to mail a letter with given weight in Danmania.

    >>> coins_required(100)
    80
    >>> coins_required(101)
    105
    ""

    if weight <= 30:
        return 40
    elif weight <= 50:
        return 55
    else:
        excess = weight - 50
        return 55 + math.ceil(excess / 50) * 25

    # alternate:
    # if excess % 50 == 0:
    #    return 55 + excess // 50 * 25
    # else:
    #    return 55 + (excess // 50 + 1) * 25
Question 4.  [7 marks]

For the purposes of this question, any sequence of consecutive characters that does not contain a blank character is considered to be a word. Using the constant BLANK_CHAR whenever possible, complete the function according to its docstring description.

Reminder: The following expressions are all True:
'a'.upper() == 'A',  '4'.upper() == '4',  ';'.upper() == ';',  and ' '.upper() == ' '.

BLANK_CHAR = ' '  

```python
def capitalize(s):
    
    """ (str) -> str

    Return a version of s that has the first character of every word in s converted to upper-case.

    >>> capitalize('february 11th is a GREAT day for writing a test!')
    'February 11th Is A GREAT Day For Writing A Test!'
    >>> capitalize(' 123 ')
    ' 123 '  
    """

    result = s[0].upper()
    for i in range(1, len(s)):
        if s[i-1] == BLANK_CHAR:
            result = result + s[i].upper()
        else:
            result = result + s[i]

    return result

# alternate:
# result = s[0].upper()
# prev = s[0]
# for i in range(1, len(s)):
#     if prev == BLANK_CHAR:
#         result = result + s[i].upper()
#     else:
#         result = result + s[i]
#     prev = s[i]

# return result
```