### Question 1.  [6 marks]

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>
</table>
| `course = 'CSC' + 108`  
  `print(course)` | Error, cannot concatenate str to int |
| `L = [1, 2]`  
  `L = L.append(3)`  
  `print(L)` | None |
| `for value in range(9, 1, -3):`  
  `print(value)` | 9  
  6  
  3 |
| `s = 'pi'`  
  `v = float(s)`  
  `print(v)` | Error, cannot convert non-digits to float |
| `list1 = [5, 4, 3, 2, 1]`  
  `element = list1[2:][1]`  
  `print(element)` | 2 |
| `result = 'instilling'.find('in', 1)`  
  `print(result)` | 7 |
Question 2.  [2 marks]

Complete the docstring examples with arguments that will cause the function calls to return the values shown.

def midterm_function(s: str, i: int) -> bool:
    
    """
    Precondition: len(s) >= 1 and 0 <= i < len(s)
    
    # first argument: any str with a digit at index i
    # second argument: 0 <= i < len(s)
    
    # There are many possible solutions. Here is an example:
    >>> midterm_function('416', 1)
    True
    >>> midterm_function('6ix', 2)
    False
    """
    
    return s[i].isdigit()
Question 4.  [4 marks]

Complete the following function according to its docstring.

```python
def pet_licence_fee(dogs: int, cats: int) -> int:
    """Return the pet licence fee (in dollars) for a household that has the
given number of dogs and cats, according to the following fee schedule:

<table>
<thead>
<tr>
<th>total number of dogs and cats</th>
<th>licence fee</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0 dollars</td>
</tr>
<tr>
<td>1 to 3, inclusive</td>
<td>60 dollars</td>
</tr>
<tr>
<td>over 3</td>
<td>100 dollars</td>
</tr>
</tbody>
</table>

The licence fee is doubled if there are more dogs than cats in the
household.

Precondition: dogs >= 0 and cats >= 0

>>> pet_licence_fee(1, 1)
60
>>> pet_licence_fee(3, 2)
200
>>> pet_licence_fee(2, 3)
100
"""

    total = dogs + cats
    if total == 0:
        return 0
    elif total <= 3:  # cannot instead write: if total <= 3 and did not return above
        result = 60
    else:
        result = 100

    if dogs > cats:
        result = 2 * result

    return result
```
**Question 5.** [5 marks]

Complete the following function according to its docstring.

```python
def num_upper_digits_same(s: str) -> bool:
    """Return True if and only if s contains the same number of uppercase
    letters as digits."

    count_digits = 0
    count_uppers = 0

    for ch in s:
        if ch.isdigit():  # ch in '0123456789'
            count_digits = count_digits + 1
        elif ch.isupper():  # ch in 'ABCDE...'
            count_uppers = count_uppers + 1

    return count_digits == count_uppers
```

```python
>>> num_upper_digits_same('CSC108')
True
>>> num_upper_digits_same('COMPUTER SCIENCE 108')
False
>>> num_upper_digits_same('apple')
True
"""
```
Question 6.  [3 marks]

Fill in the box with the while loop condition required for the function to work as described in its docstring.

```python
def find_lowercase_vowel(msg: str) -> int:
    """Return the index of the first lowercase vowel (a, e, i, o, u) in msg, or the length of msg if it does not contain any lowercase vowels.
    >>> find_lowercase_vowel('cats')
    1
    >>> find_lowercase_vowel('python')
    4
    >>> find_lowercase_vowel('AbcdE')
    5
    """

    i = 0
    while i < len(msg) and msg[i] not in 'aeiou':
        i = i + 1
    return i
```