**Question 1.**  [8 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>
<tbody>
<tr>
<td><code>print(type(20 / 4) == type(10 // 2))</code></td>
<td>False</td>
</tr>
<tr>
<td><code>print(10 / 5 * 2)</code></td>
<td>4.0</td>
</tr>
</tbody>
</table>
| `words = ['best', 'of', 'luck']
words.append(words[0])
print(words)`             | ['best', 'of', 'luck', 'best']           |
| `values = [5, 1, 4]
values.extend([8])
print(values)`             | [5, 1, 4, 8]                            |
| `print('apple' < 'banana' and True)`                                | True                                    |
| `sunny = True
num = 0
print(10 / num == 0 or sunny)`                                      | ERROR Division by zero                   |
| `print('hi' + 5)`                                                    | ERROR! Cannot concatenate string with int. |
| `a_list = [13, 4, 7]
b_list = a_list + [10]
print(b_list)`           | [13, 4, 7, 10]                          |
**Question 2.** [4 marks]

In the function below, complete (i) the function description in the space provided, and (ii) the example function calls by adding arguments that result in the return values shown. (For the example calls, there may be several correct answers, and providing any one of them will earn full marks.)

```python
def mystery(message):
    """(str) -> str
    Precondition: len(message) >= 1
    Return message iff the first character is a vowel, otherwise return a string with every other character.
    >>> mystery('apple')
    'apple'
    >>> mystery('banana')
    'bnn'
    """
    if message[0].lower() in 'aeiou':
        return message
    else:
        return message[::2]
```

```python
>>> mystery('apple')
'apple'
>>> mystery('banana')
'bnn'
```
Question 3. [4 marks]

Read the function header and function body, and then complete the docstring. Write the type contract and the description, and give two examples that return different values. Preconditions are not required.

def hidden_function_name(s):
    """ (str) -> bool

    Return True iff number contains only digits and '-', '(', and ')'.

    >>> hidden_function_name('(416)-123-4567')
    True
    >>> hidden_function_name('123-ABCD')
    False

    """

    i = 0
    while i < len(s):
        char = s[i]
        if not (char.isdigit() or char in '()-'):
            return False
        i = i + 1
    return True
Question 4.  [5 marks]

Complete this function according to its docstring description.

```python
def get_greater_string(message, char):
    """ (str, str) -> str
    
    Precondition: len(char) == 1 and
      (message.isalpha() and message.islower()) == True and
      (char.isalpha() and char.islower()) == True
    
    Return a string that contains all characters from message that appear later
    in the alphabet than char, in the order they appear in message.
    
    >>> get_greater_string('watermelon', 'g')
    'wtrmlon'
    >>> get_greater_string('banana', 'x')
    ''
    """

    result = ''
    for ch in message:
        if ch > char:
            result = result + ch
    return result
```

Question 5.  [3 marks]

Complete this function according to its docstring description.

```python
def get_string_info(message, char):
    """ (str, str) -> bool

    Precondition: len(message) > 0 and len(char) == 1 and
                   (char in message == True) and
                   message contains an odd number of characters

    Return True if and only if the first occurrence of char in message is
    after the middle index in message.

    >>> get_string_info('hello', 'l')
    False
    >>> get_string_info('pythons', 'o')
    True
    """

    return message.find(char) > len(message) // 2
```