**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>print(3 + 5 % 2)</td>
<td>4</td>
</tr>
<tr>
<td>num = len([10, 8, 2]) print(num)</td>
<td>3</td>
</tr>
<tr>
<td>values = [3, 7, 2] values[-1] = 'random' print(values)</td>
<td>[3, 7, 'random']</td>
</tr>
<tr>
<td>word = 'good luck' word[0] = 'G' print(word)</td>
<td>ERROR. Strings are immutable.</td>
</tr>
<tr>
<td>print(False or ('october' &lt; 'november'))</td>
<td>False</td>
</tr>
<tr>
<td>snowing = False num = 0 print(snowing and 5 / num == 0)</td>
<td>False</td>
</tr>
<tr>
<td>word = 'Code!' print(word[5])</td>
<td>ERROR string index out of range</td>
</tr>
<tr>
<td>a = [13, 4, 7] b = a.append(5) print(a)</td>
<td>[13, 4, 7, 5]</td>
</tr>
</tbody>
</table>
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

    Return a string with every third character in message if the first character
    is a digit, otherwise return message.

    >>> mystery('24cat')
    '2a'
    >>> mystery('hello')
    'hello'
    """
    if message[0].isdigit():
        return message[::3]
    else:
        return message
```

```python
g>>> mystery('24cat')
g'2a'
g>>> mystery('hello')
g'hello'
g```
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.

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

    Return True iff s contains only letters, digits, '@' and '.'.

    >>> hidden_function_name('jsmith1@cs.toronto.edu')
    True
    >>> hidden_function_name('jsmith at cs dot toronto dot edu!')
    False
    """

    i = 0
    while i < len(s):
        char = s[i]
        if not (char.isalpha() or 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 avoid_characters(message, avoid_string):
    """ (str, str) -> str

    Return a string containing all characters in message that are not in the
    avoid_string, in the order they appear in message.
    
    >>> avoid_characters('Hello world!', 'deer')
    'Hllo wol!'
    >>> avoid_characters('banana', 'dog')
    'banana'
    """

    result = ''
    for char in message:
        if char not in avoid_string:
            result += char
    return result
```

```python
>>> avoid_characters('Hello world!', 'deer')
'Hllo wol!'
>>> avoid_characters('banana', 'dog')
'banana'
```
Question 5. [3 marks]

Complete this function according to its docstring description.

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

    Precondition: len(char) == 1 and message contains char

    Return the rest of message after the first occurrence of char.
    
    >>> get_string_info('watermelon', 'r')
    'melon'
    >>> get_string_info('apple', 'p')
    'ple'
    """

    return message[message.find(char) + 1:]
```

```