Question 1.  [4 marks]

Part (a)  [1 mark] In the box below, write what this code prints:

```python
k = 3
w = k
k = k + 1
print(k, w)
```

Solution:
4 3

Part (b)  [1 mark] In the box below, write what this code prints:

```python
s = 'substitute'
p = len(s)
i = len(s) // 2
while i <= len(s):
    print(s[i - 1])
    i = i + 2
```

Solution:
10
t
t
t

Part (c)  [2 marks] For each code fragment below, fill in the table with what it prints.

```python
s1 = 'elephant'
s2 = 'giraffe'
```

<table>
<thead>
<tr>
<th>Code</th>
<th>What is printed</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>print(s1[0] + s2[1])</code></td>
<td><code>ei</code></td>
</tr>
<tr>
<td><code>print(s2[2:4] + s1[3:5])</code></td>
<td><code>raph</code></td>
</tr>
<tr>
<td><code>print(abs(len(s2) - len(s1)))</code></td>
<td>1</td>
</tr>
<tr>
<td><code>print(not (s2 in s1))</code></td>
<td><code>True</code></td>
</tr>
</tbody>
</table>
Question 2.  [3 marks]

A store is having a sale. A 10% discount is given for purchases from $50 (inclusive) to $100 (exclusive), and a 20% discount for purchases $100 and up.

Complete the following function according to the description above and its docstring below. Use the constants when appropriate.

DISCOUNT1 = 0.10
DISCOUNT2 = 0.20

def apply_discount(dollar_amount):
    """ (float) -> float
    Precondition: dollar_amount >= 0
    Return dollar_amount with the appropriate discount applied."

    >>> apply_discount(25.5)
    25.5
    >>> apply_discount(75.0)
    67.5
    >>> apply_discount(120.0)
    96.0
    """

SOLUTION:

    if dollar_amount >= 100:
        return dollar_amount - (dollar_amount * DISCOUNT2)
    elif dollar_amount >= 50:
        return dollar_amount - (dollar_amount * DISCOUNT1)
    else:
        return dollar_amount
Question 3.  [11 MARKS]

Part (a)  [5 MARKS] Write the body of the following function according to its docstring description.

```python
def sort_corresponding(s1, s2):
    """ (str, str) -> tuple of (str, str)"
    
    Precondition: len(s1) == len(s2)
    
    Return a tuple of two strings: the characters of s1 that are the same as the characters at the corresponding position of s2, and the characters of s1 that are different from the characters at the corresponding position of s2.
    
    >>> sort_corresponding('hello!', 'heyya!')
    ('he!', 'llo')
    >>> sort_corresponding('bunk bed', 'buttered')
    ('bued', 'nk b')
    """

SOLUTION:

same = ''
different = ''

for i in range(len(s1)):
    if s1[i] == s2[i]:
        same = same + s1[i]
    else:
        different = different + s1[i]

return (same, different)
```
Part (b) [3 marks] Write the body of the following function according to its docstring description. Do not use any str methods.

```python
def greater_than(s, d):
    """ (str, int) -> bool

    Precondition: s contains only digits. d is a digit.
    Return True iff s contains a digit greater than d.
    >>> greater_than('1234', 3)
    True
    >>> greater_than('123', 5)
    False
    """

    for ch in s:
        if int(ch) > d:
            return True

    return False
```

SOLUTION:
Part (c) [3 marks] Read the function header and body and then complete the docstring, including the type contract, description, and two examples that return different values.

```python
def mystery(s, ch):
    """ (str, str) -> bool

    Precondition: len(s) >= 1 and len(ch) == 1
    Return True iff the first character of s is ch, ignoring case.

    >>> mystery('hello', 'h')
    True
    >>> mystery('hello', 's')
    False
    """
```