1. Work through the entire execution of the following program, show the program’s output, and draw the stack as it exists just before the return statement in `mystery` is executed.

```python
def mystery(L1, L2):
    count = 0
    for i in range(len(L1)):
        if L1[i] == L2[i]:
            L2[i] = L2[i].upper()
            count = count+1
        else:
            L2[i] = "-"
    #draw stack here
    return count

def main():
    word1 = "pets"
    word2 = "pass"
    ls1 = list(word1)
    ls2 = list(word2)
    val = mystery(ls1, ls2)
    print(word2)
    print(ls2)
    print(val)

main()
```

**PROGRAM OUTPUT**

---
2. Show what the low, high, and mid index values are for each step in searching the list $L$ for the value 15 using a binary search. Also show $L[mid]$ for each step. The indices for each item in the list are shown below each element in $L$.

$x = 15$
$L = [-45, -30, 0, 10, 15, 32, 48, 50, 65, 70, 77]$

<table>
<thead>
<tr>
<th>index</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>low</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>mid</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>high</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$L[mid]$</td>
<td>-45</td>
<td>-30</td>
<td>0</td>
<td>10</td>
<td>15</td>
<td>32</td>
<td>48</td>
<td>50</td>
<td>65</td>
<td>70</td>
<td>77</td>
</tr>
</tbody>
</table>

3. If there are $n = 16$ items in a list, what is the maximum number of steps/comparisons linear and binary search will each need to lookup a value in the list? What is the minimum number of steps/comparisons each search will need?

4. Write a function called `countShort` that has two parameters, a string containing zero or more words separated by spaces, and an integer value $val$. The function should return the number of words whose length is less than or equal to $val$.

For example, `countShort("a bee sees De", 3)` should return 3 while `countShort("Swarthmore College Computer Science", 5)` should return 0.