1. An “abstract” view of the stack:

Using an array implementation would look something like:

<table>
<thead>
<tr>
<th>items:</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>(max-1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>b</td>
<td>c</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>top:</th>
<th>max: 100</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>

Complete the big-oh notation for the following stack methods assuming an array implementation: ("n" is the # items)

<table>
<thead>
<tr>
<th>Big-oh</th>
<th>push(item)</th>
<th>pop()</th>
<th>peek()</th>
<th>size()</th>
<th>isEmpty()</th>
<th>isFull()</th>
<th>Constructor</th>
</tr>
</thead>
</table>

2. Since Python does not have a (directly accessible) built-in array, we can use a list.

```python
class Stack:
    def __init__(self):
        self.items = []

    def isEmpty(self):
        return self.items == []

    def push(self, item):
        self.items.append(item)

    def pop(self):
        return self.items.pop()

    def peek(self):
        return self.items[len(self.items)-1]

    def size(self):
        return len(self.items)
```

Since Python uses an array of references (pointers) to list items in their implementation of a list.

a) Complete the big-oh notation for the stack methods assuming this Python list implementation: ("n" is the # items)

<table>
<thead>
<tr>
<th>Big-oh</th>
<th>push(item)</th>
<th>pop()</th>
<th>peek()</th>
<th>size()</th>
<th>isEmpty()</th>
<th><strong>init</strong></th>
</tr>
</thead>
</table>

b) Which operations should have what preconditions?
3. The text’s alternative stack implementation also using a Python list is:

```python
class Stack:
    def __init__(self):
        self.items = []

    def isEmpty(self):
        return self.items == []

    def push(self, item):
        self.items.insert(0, item)

    def pop(self):
        return self.items.pop(0)

    def peek(self):
        return self.items[0]

    def size(self):
        return len(self.items)
```

Since an array is used to implement a Python list, the alternate Stack implementation using a list:

```
<table>
<thead>
<tr>
<th>&quot;Abstract&quot; Stack</th>
<th>&quot;alternate&quot; Stack Object</th>
</tr>
</thead>
<tbody>
<tr>
<td>c top</td>
<td>items:</td>
</tr>
<tr>
<td>b</td>
<td>c b a</td>
</tr>
<tr>
<td>a bottom</td>
<td></td>
</tr>
</tbody>
</table>
```

a) Complete the big-oh notation for the “alternate” Stack methods: ("n" is the # items)

<table>
<thead>
<tr>
<th></th>
<th>push(item)</th>
<th>pop()</th>
<th>peek()</th>
<th>size()</th>
<th>isEmpty()</th>
<th><strong>init</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Big-oh</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

4. How could we use a stack to check if a word is a palindrome (e.g., radar, toot)?

5. How could we check to see if we have a balanced string of nested symbols? ("(([]{}))[]")