Question 1. Write a recursive function called addch that has two parameters: a string `S` and a character `ch`. The function should return a string with the character added between each letter of the original string. For example, calling `addch("hello", "*")` would return "h*e*l*l*o".

Question 2. Assume the mystery function is originally called as follows: `mystery("EVIL",1)`.

- Draw the stack as it would look at it’s largest
- Show what is printed as the functions recur
- Show what is finally returned by the first call to `mystery()`

```python
def mystery(S,n):
    STACK:
    print("%2d: %s" % (n, S))
    if len(S) == 0:
        return S
    else:
        fch = S[0]
        rest = S[1:]
        result = mystery(rest,n+1) + fch
        return result
```

Functions print:

```
What is finally returned?
```
Question 3. For each of the algorithms below, attach the correct label: $O(n \log n)$, $O(n^2)$, $O(\log n)$, and $O(n)$

binary search:

merge sort:

bubble sort:

linear search:

Question 4. Imagine you are writing a playlist class for some music application, like iTunes or Spotify. Create a Playlist class that works with the following test code, and produces the results below. You should write the \_init\_, \_str\_, and addSong methods. For the constructor, only the name of the playlist is given. For the addSong method, the name of the song is given.

```python
p = Playlist("Workout Music")
p.addSong("Pump It")
p.addSong("Back In Black")
p.addSong("We Will Rock You")
p.addSong("Panama")
print(p)
```

Playlist: Workout Music
1: Pump It
2: Back In Black
3: We Will Rock You
4: Panama