Recursion Cheatsheet

Use this algorithm when:

- Unlike the other algorithms, unfortunately there is no exact criteria as to when we can use recursion, so use your best guess, but as a rule of thumb use recursion when:
- You can always make one “step” in a problem which gets you closer to the answer, but doesn’t get you to the answer directly

Steps to using the algorithm:

Thinking about the problem:

1) Define the state
    a) This is something that uniquely defines the answer. If I give you the state, there should only be one possible answer.
2) Figure out the base cases
    a) These are the states where finding the answer is really easy
3) Figure out the transitions
    a) This is how you move from one state to another
    b) Make sure that the state gets a little bit closer to the base case every time you transition. If not, you may never reach the base case and have infinite recursive calls.

Implementation:

```
[return type] function_name (state) {
     if statement for the base case {
          return the answer to the base case
     }
     transitions
}
```

How to calculate the runtime:

Draw out a diagram for the recursive calls and from the diagram, calculate the number of recursive calls. The number of recursive calls is the runtime.