

Thinking Recursively

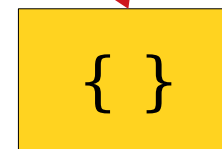
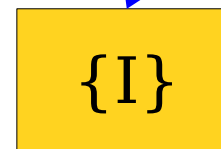
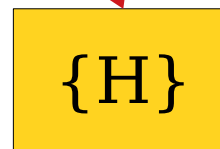
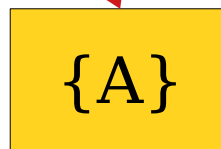
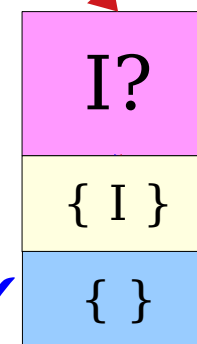
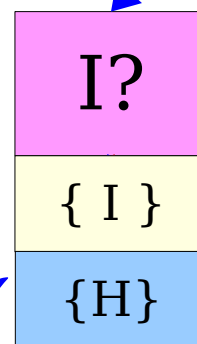
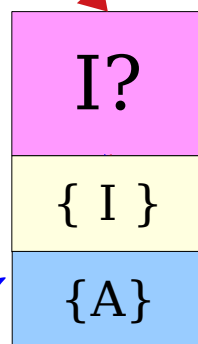
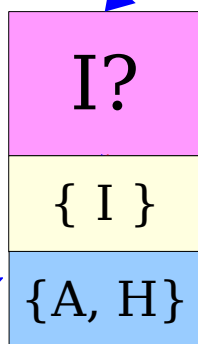
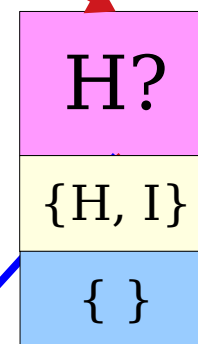
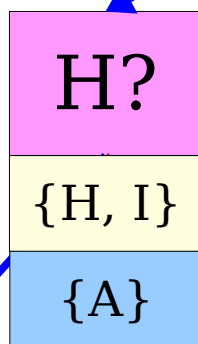
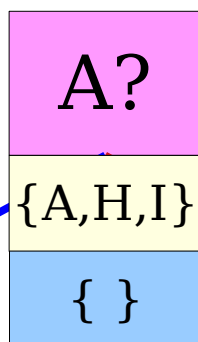
Part III

Outline for Today

- ***Iteration + Recursion***
 - Combining two techniques together.
- ***Enumerating Permutations***
 - What order should we do things?
- ***Enumeration, Generally***
 - How to think about enumeration problems.

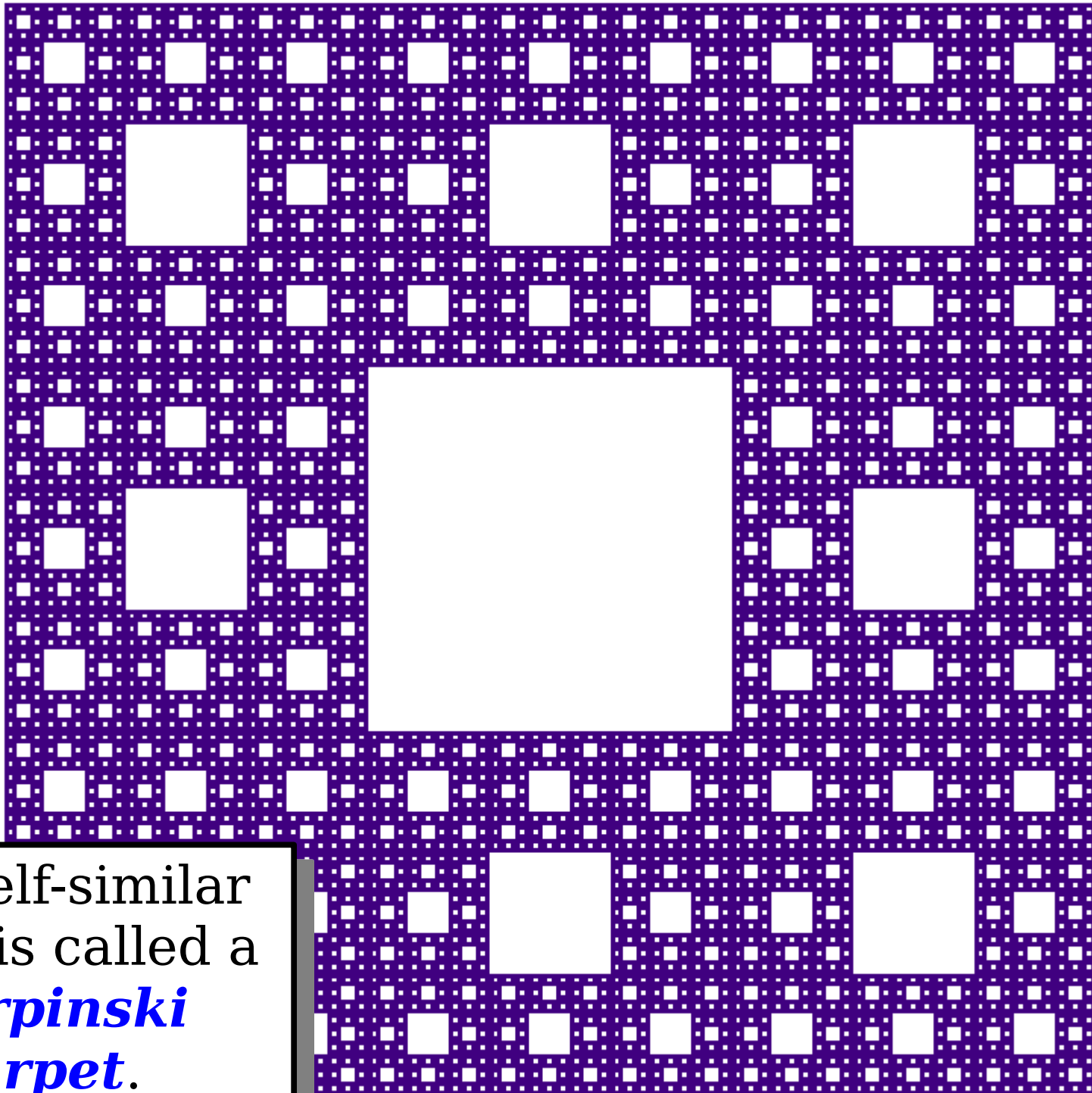
Recap from Last Time

List all *subsets* of
 $\{A, H, I\}$

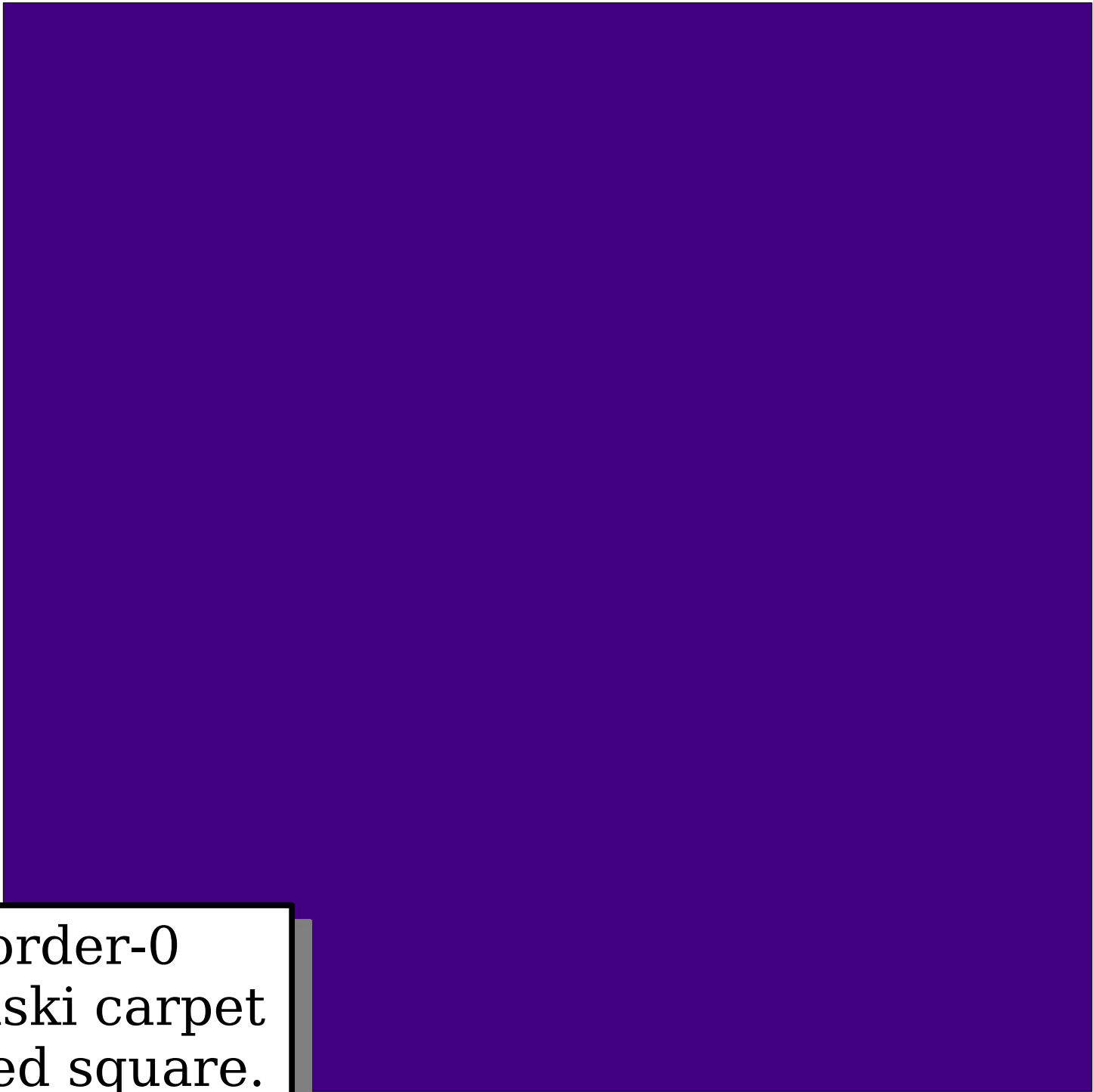


New Stuff!

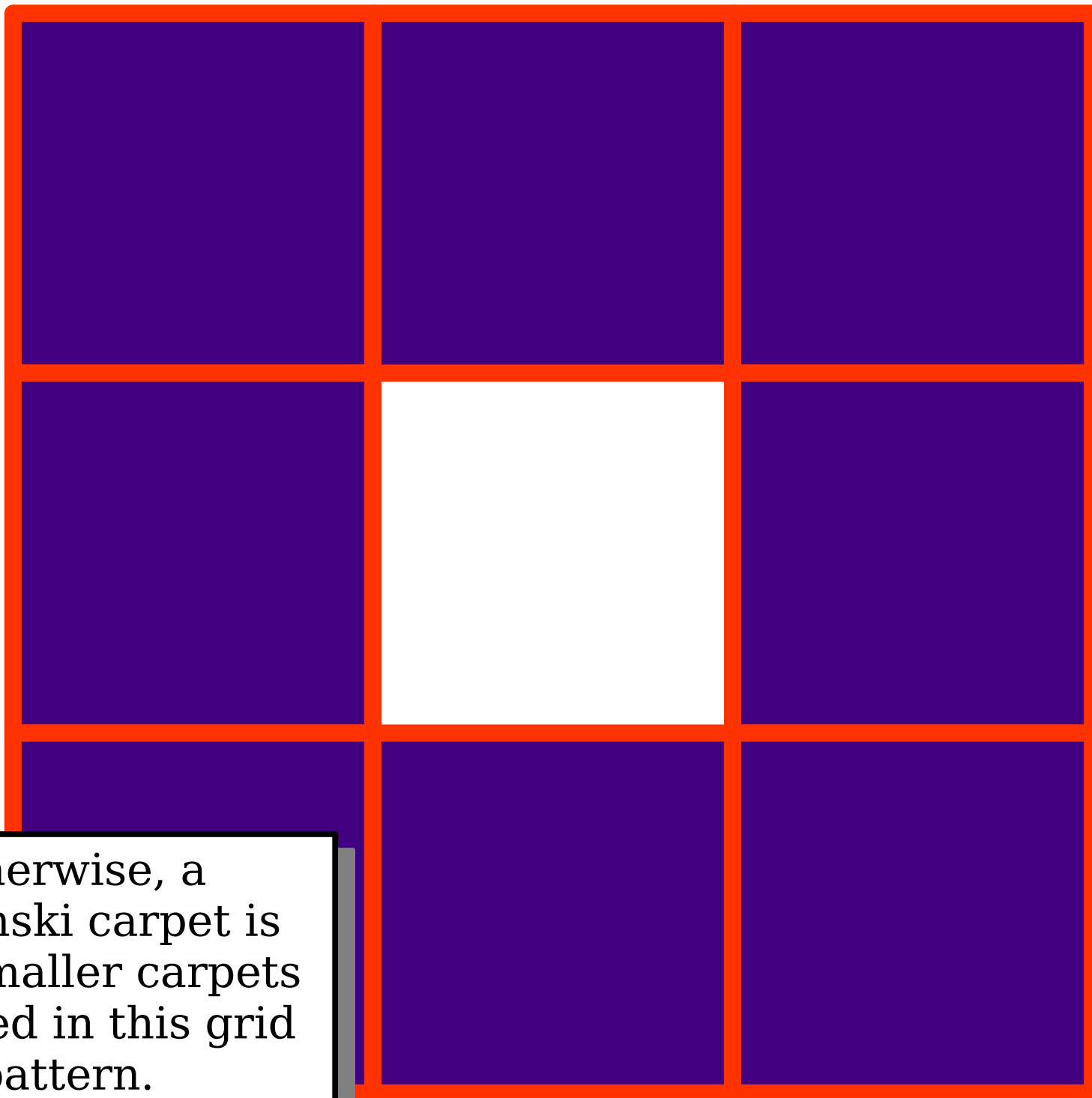
More On Self-Similarity



This self-similar shape is called a ***Sierpinski carpet.***



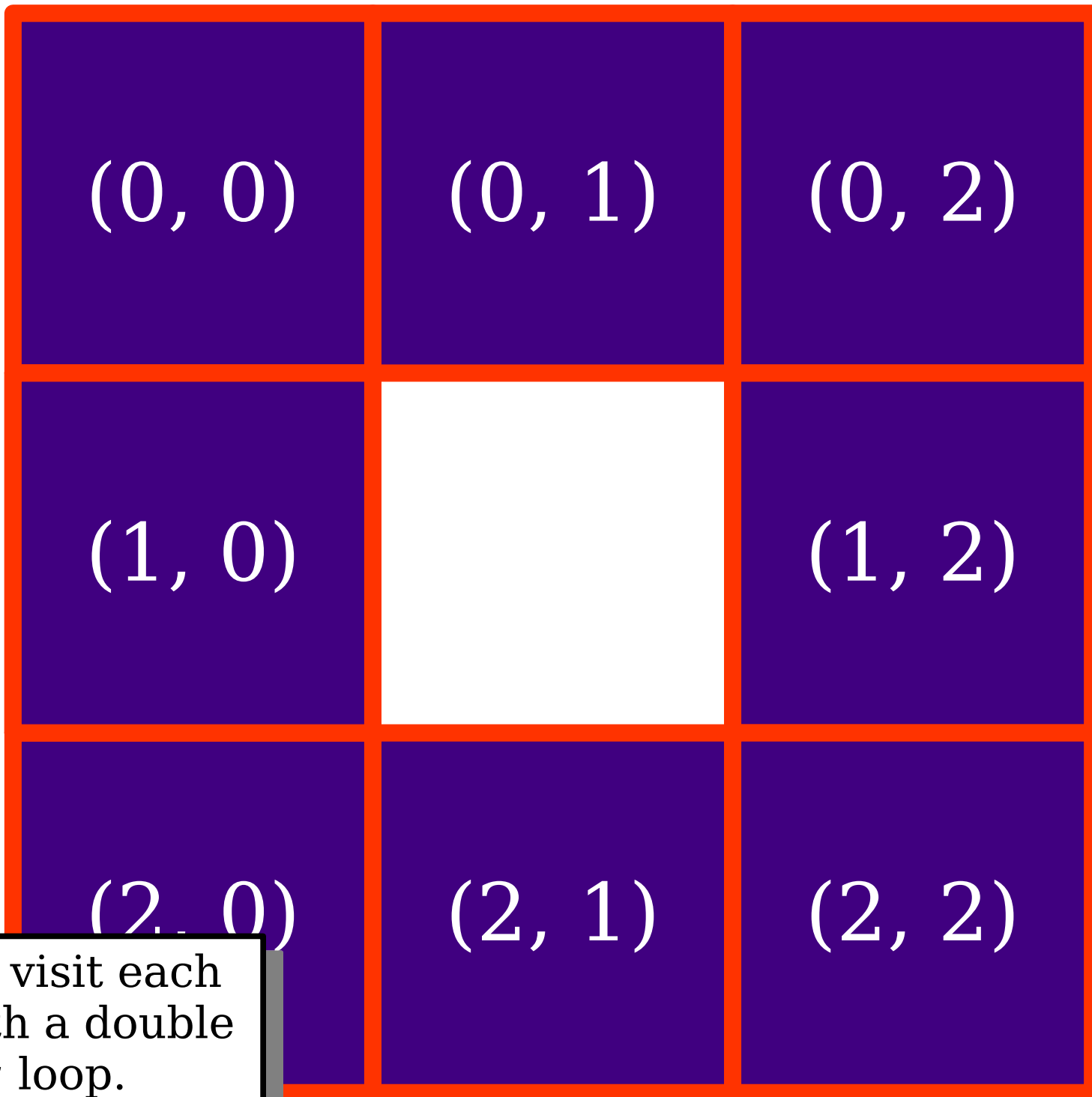
An order-0
Sierpinski carpet
is a filled square.



Otherwise, a Sierpinski carpet is eight smaller carpets arranged in this grid pattern.

$(0, 0)$	$(0, 1)$	$(0, 2)$
$(1, 0)$		$(1, 2)$
$(2, 0)$	$(2, 1)$	$(2, 2)$

Label each square
with its (row, col).



We can visit each spot with a double for loop.

Iteration + Recursion

- It's completely reasonable to mix iteration and recursion in the same function.
- Here, we're firing off eight recursive calls, and the easiest way to do that is with a double for loop.
- Recursion doesn't mean "the absence of iteration." It just means "solving a problem by solving smaller copies of that same problem."

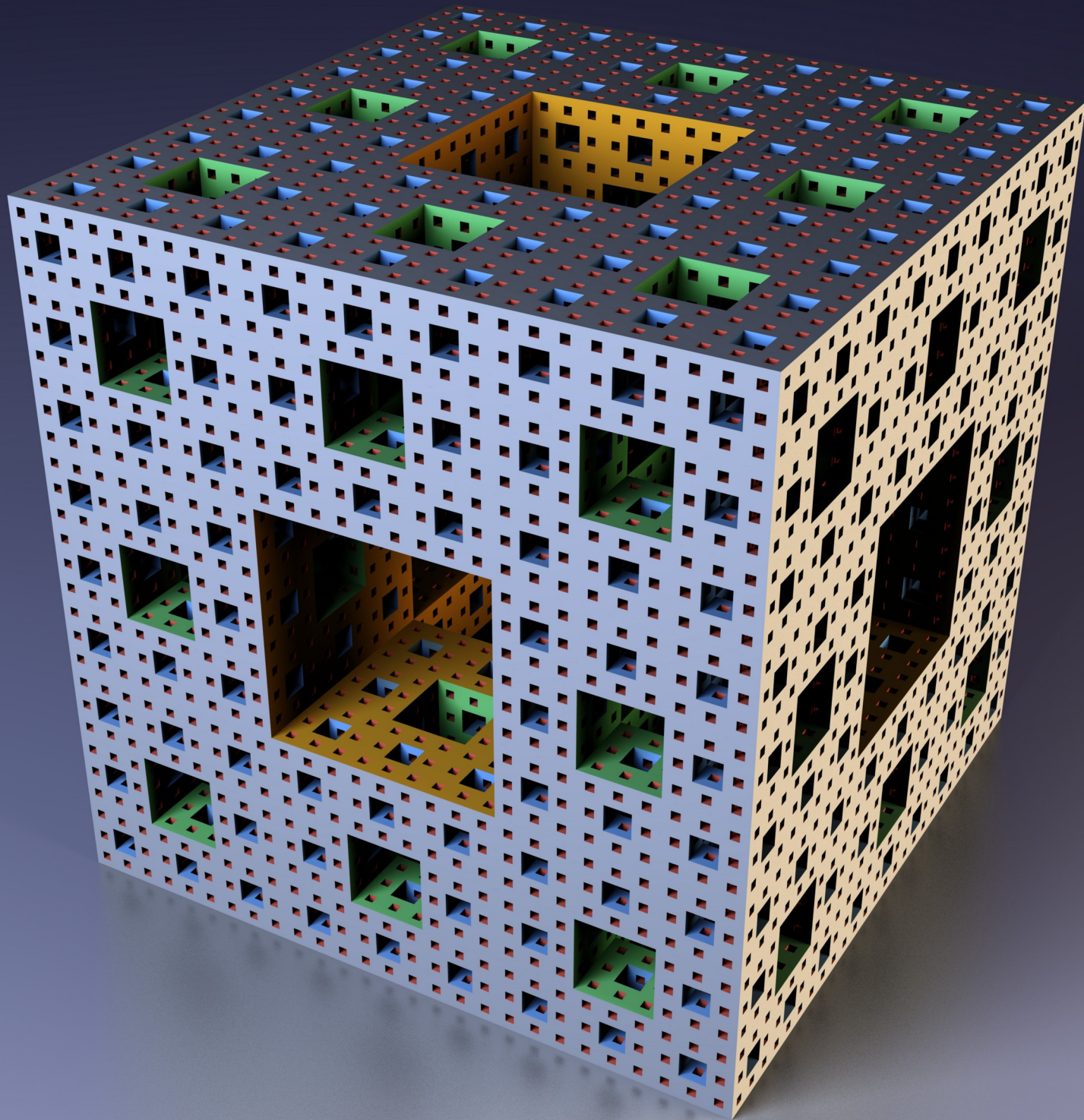
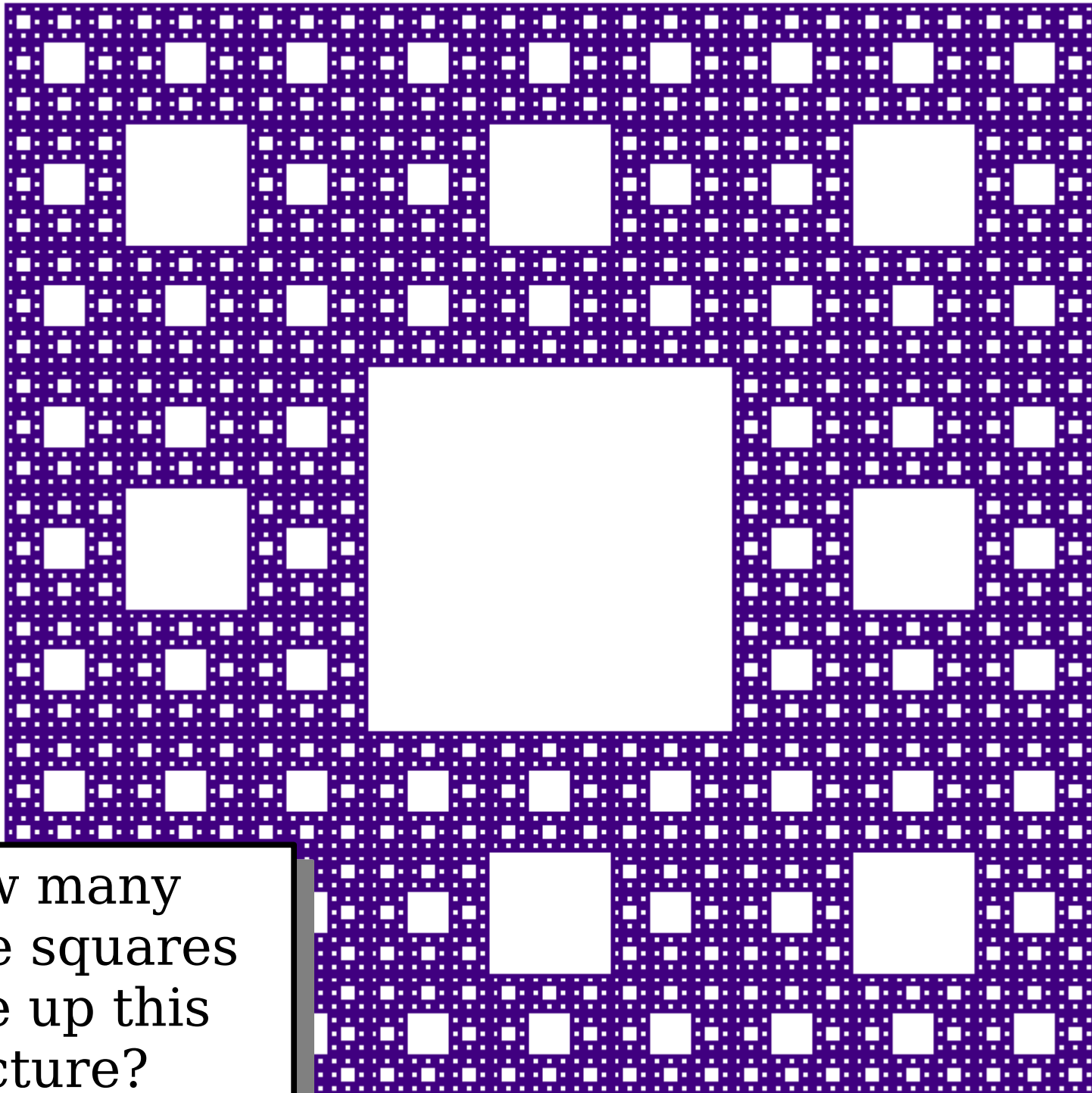


Image Credit



How many
purple squares
make up this
picture?

Time-Out for Announcements!

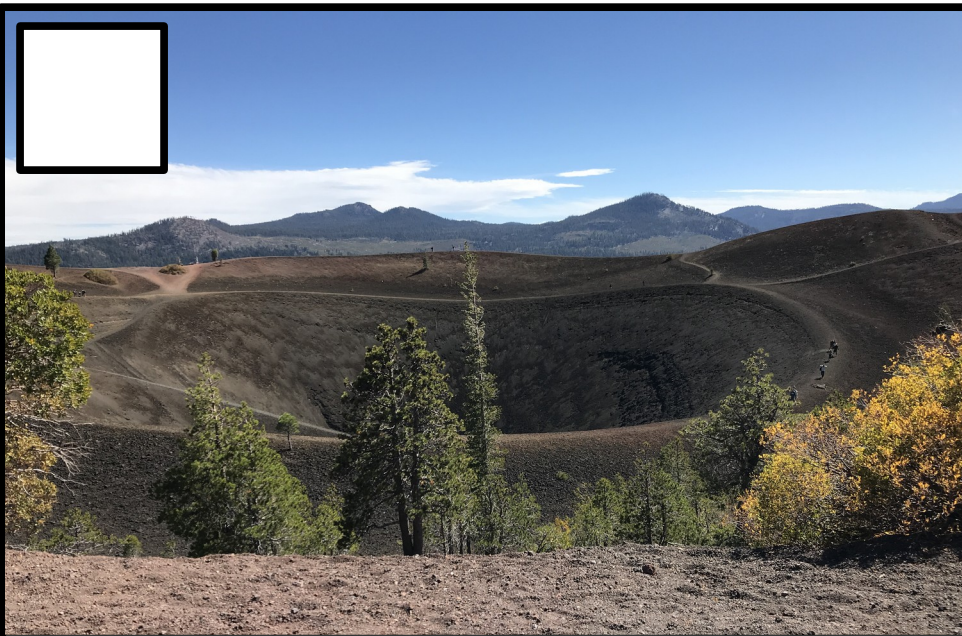
Assignment 3

- Assignment 3 went out last Friday and is due this Friday at 10:30AM.
- You're welcome to work in pairs, as long as your partner is also in your section.
- If you're following our recommended timetable, at this point you'll have finished the Sierpinski triangle and made some progress on Human Pyramids.
- Aim to complete "What Are YOU Doing?" by next lecture and to start Shift Scheduling.

(The Curtain Rises on Act II)

Enumerating Permutations

A ***permutation*** is a rearrangement of the elements of a sequence.



Lassen Volcanic National Park



Yosemite National Park



Joshua Tree National Park



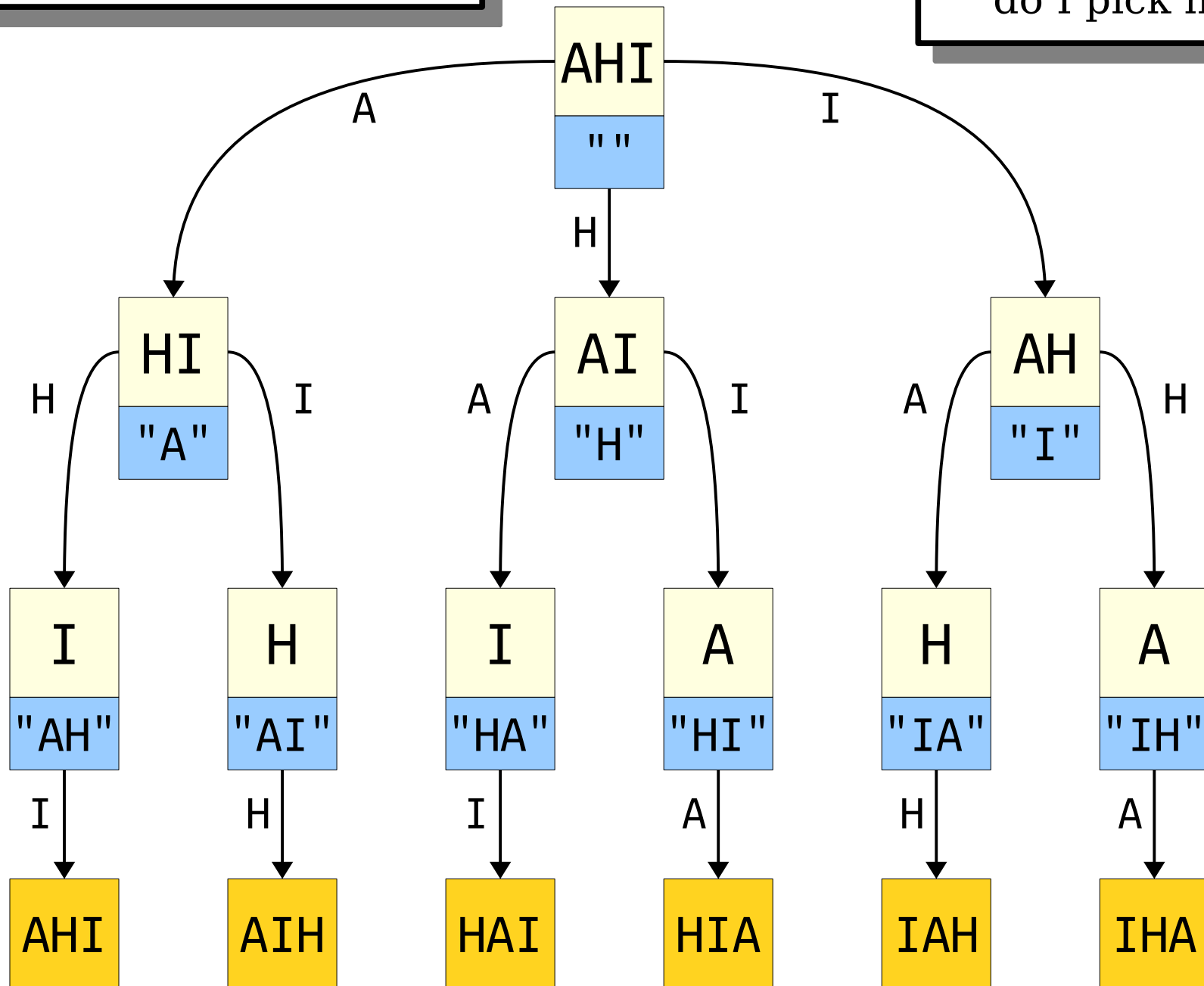
Lava Beds National Monument

List all *permutations* of
 $\{A, H, I\}$

Each decision is of
the form “which item
do I pick next?”

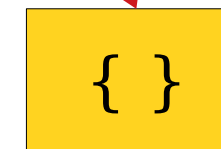
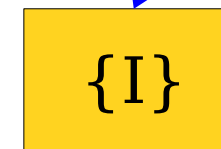
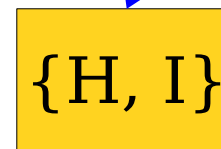
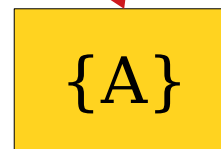
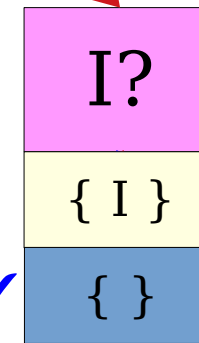
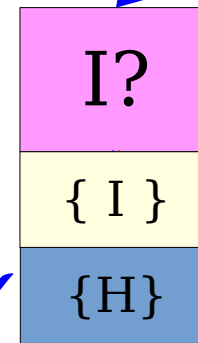
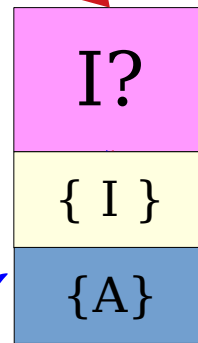
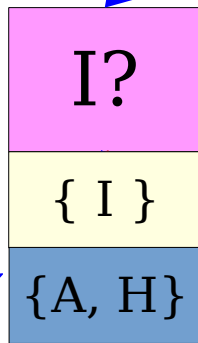
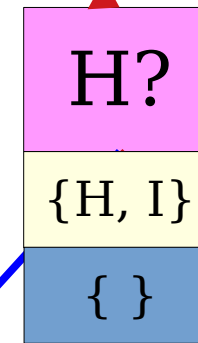
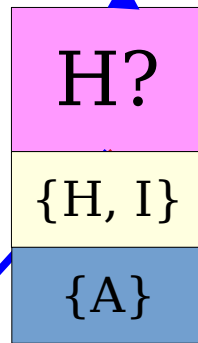
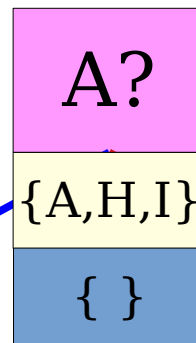
List all *permutations* of
{A, H, I}

Each decision is of
the form "which item
do I pick next?"



List all *subsets* of
 $\{A, H, I\}$

Each decision is of
the form “do I
include this item?”



Storing Permutations


```
Set<string> permutationsOf(const string& str);
```

Base Case: No decisions remain.

```
ResultType exploreRec(decisions remaining,  
                      decisions already made) {
```

```
  if (no decisions remain) {  
    return decisions made;  
  } else {
```

```
    ResultType result;
```

```
    for (each possible next choice) {  
      result += exploreRec(all remaining decisions,  
                          decisions made + that choice);
```

```
    }  
    return result;
```

```
  }  
}
```

Recursive Case:

Try all options for the next decision.

```
ResultType exploreAllTheThings(initial state) {  
  return exploreRec(initial state, no decisions made);  
}
```

Summary for Today

- Recursion and iteration aren't mutually exclusive and are frequently combined.
- We can enumerate subsets using a decision tree of “do I pick this?” We can enumerate permutations using a decision tree of “what do I pick next?”
- Recursive functions can both print all objects of some type and return all objects of some type.

Your Action Items

- ***Read Chapter 8***
 - There are so many goodies there, and it's a great way to complement what we're discussing here.
- ***Work on Assignment 3***
 - Aim to complete What Are YOU Doing? by Wednesday and start working on Shift Scheduling.

Next Time

- ***Enumerating Combinations***
 - Can you build the Dream Team?
- ***Recursive Backtracking***
 - Finding a needle in a haystack.
- ***The Great Shrinkable Word Problem***
 - A fun language exercise with a cute backstory.