To partition a class $C$ is to separate it into non-empty subclasses in such a way that every member of $C$ is a member of one and only one of the subclasses. For example, the class $\{1, 2, 3, 4, 5\}$ can be partitioned into two subclasses, respectively comprising its even and odd members: $\{1, 3, 5\}$ and $\{2, 4\}$. The class of those subclasses—in this case, $\{\{1, 3, 5\}, \{2, 4\}\}$—is called a partition of $C$.

Usually, it is possible to partition a class in many different ways. The class $\{1, 2, 3, 4, 5\}$ can be partitioned in no fewer than fifty-two ways, if you include both the partition $\{\{1\}, \{2\}, \{3\}, \{4\}, \{5\}\}$, in which each member gets a subclass of its own, and the partition $\{\{1, 2, 3, 4, 5\}\}$, in which all the members wind up in the same subclass.

If a class has only one member, say $a$, there is only one partition of it: $\{\{a\}\}$. (Recall that the empty class cannot belong to a partition.)

You might think it impossible to form a partition of the empty class, but in fact there is one such partition, namely the empty class itself. (It is “vacuously true” that every member of the empty class belongs to one and only one member of the empty class!)

For a family $F$ of classes to be a partition of a class $C$, the members of $F$ must all be disjoint from one another, and their union must be equal to $C$. Any class of disjoint non-empty classes is a partition of the union of all of its members.

Let’s try to work out a recursive procedure for generating the class of all partitions of a given class $C$. The base case, where $C$ is empty, has already been addressed: There is only one partition of $\emptyset$, namely $\emptyset$ itself, so the class of all the partitions of $\emptyset$ is $\{\emptyset\}$.

Suppose, then, that $C$ is not empty, so that we can choose and designate some member $a$ of $C$ to work with, and apply our procedure recursively on $C - \{a\}$ to obtain the class of all partitions of that slightly smaller class. How can we combine $a$ with the class of all partitions of $C - \{a\}$ to construct the class of all partitions of $C$?

Reasoning combinatorially, we may reflect that, in each partition of $C$, our designated member $a$ will either be alone in a subclass (so that $\{a\}$ is
one member of the partition), or it will join one of the subclasses within a partition of \( C - \{a\} \). So there are two recipes for constructing a partition of \( C \). The first is to take a partition of \( C - \{a\} \) and add the new subclass \( \{a\} \) to that partition. The second is to take a partition of \( C - \{a\} \), choose one of the subclasses within that partition, and add \( a \) to that subclass.

As an illustration, let’s suppose that \( C \) is the class \( \{0, 1, 2\} \), and let’s arbitrarily choose 2 as our designated member \( a \), so that \( C - \{a\} \) is \( \{0, 1\} \). It turns out that there are two partitions of \( \{0, 1\} \) — the one in which the two values wind up in different subclasses, and the one in which they wind up in the same subclass. So recursive application of our procedure to \( \{0, 1\} \) will yield the class comprising those two partitions, which is

\[
\{\{0\}, \{1\}\}, \{\{0, 1\}\}\}.
\]

Our first construction recipe tells us to take each of these partitions of \( \{0, 1\} \) in turn and to add \( \{2\} \) to it, obtaining \( \{\{0\}, \{1\}, \{2\}\} \) from the first partition and \( \{\{0, 1\}, \{2\}\} \) from the second.

For the other construction recipe, we must walk through each partition and consider adding the new member 2 to each subclass in turn within that partition. The first partition gives us two new partitions: \( \{\{0, 2\}, \{1\}\} \) and \( \{\{0\}, \{1, 2\}\} \). The second offers only one subclass and so yields only one new partition, \( \{\{0, 1, 2\}\} \).

Collecting all of these, we get the class containing the five partitions of \( \{0, 1, 2\} \), namely,

\[
\{\{0\}, \{1\}, \{2\}\}, \{\{0, 1\}, \{2\}\}, \{\{0, 2\}, \{1\}\}, \{\{0\}, \{1, 2\}\}, \{\{0, 1, 2\}\}\}.
\]

The construction is complete, because each partition that has \( \{2\} \) as one of its members is derived from exactly one of the partitions of \( \{0, 1\} \) by the first recipe, and any other partition is derived from exactly one of the partitions of \( \{0, 1\} \) by the second recipe.

In understanding and implementing this construction, it is important to distinguish values of four different types: the members of \( C \) (in the example, the numbers 0, 1, and 2); classes of members of \( C \) (such as \( C \) itself and its subclasses); classes of classes of members of \( C \) (such as partitions); and classes of partitions (i.e., classes of classes of classes of members of \( C \)). The construction algorithm builds and uses values of all four types.
Exercises

1. Use the construction algorithm presented above to generate the class of all partitions of \{\text{red, green, blue, yellow}\}.

2. Extend the concept of a partition so that it applies to bags. Can we adapt the construction algorithm so that it generates the class of all partitions of a given bag? How?

3. Implement the algorithm for constructing the class of all partitions of a given class in Scheme.