PART I

Counting and Listing

Enumerative combinatorics deals with listing and counting the elements in finite sets. Why is this of interest? Determining the number of elements in a finite set is the fundamental tool in the analysis of the running times and space demands of computer algorithms. It is also of importance in various areas of science, most notably statistical mechanics and structural chemistry, and in some areas of mathematics. Listing elements plays a role in the design of algorithms and in structural chemistry as well as other areas. In addition, the questions may be interesting in themselves; that is, some people find questions of the form “How many structures are there such that . . . ?” interesting.

In this part we'll study some fundamental counting and listing techniques. These tools are useful throughout combinatorics and many of them are essential for other topics that are covered later in the text. In particular, the Rules of Sum and Product form the basis of practically all of our counting calculations. The set theoretic notation and terminology that we introduce in the first two chapters is also important for the remainder of the text.

Here are some examples of the types of problems that our tools will enable us to solve systematically.

Example 1 Birthdays There are 30 students in a classroom. What is the probability that all of them have different birthdays? Let's assume that people are equally likely to be born on each day of the year and that a year has 365 days. (Neither assumption is quite correct.) Suppose we could determine the number of possible ways birthdays could be assigned to 30 people, say N, and the number of ways this could be done so that all the birthdays are different, say D. Our answer would be D/N. □

Example 2 Names In a certain land on a planet in a galaxy far away the alphabet contains only 5 letters which we will transliterate as A, I, L, S and T in that order. All names are 6 letters long, begin and end with consonants and contain two vowels which are not adjacent to each other. Adjacent consonants must be different. How many possible names are there? Devise a systematic method for listing them in dictionary order. The list begins with LALALS, LALALT, LALASL, LALAST, LALATL, LALATS, LALILS and ends with TSITAT, TSITIL, TSITIS, TSITIT. □

Example 3 Data storage An ecologist plans to simulate on a computer the growth of a biological community containing 6 competing species. He plans to try numerous variations in the environment. After each simulation, he'll order the species from most abundant to least abundant. He wants to keep track of how often each ordering occurs and, after his simulations are over, manipulate the collection of counts in various ways. How can he index his storage in a compact manner? □

Example 4 Arrangements We have 32 identical dominoes with no marks on them and a chessboard. Each domino will exactly cover two squares of the chessboard. How many ways can we arrange the dominoes so that they cover the entire board? □

Example 5 Symmetries We have a cube, some red paint and some green paint. How many different ways can we paint the cube so that each face is either all red or all green? Obtain a list of all the different ways. A first approach to this problem might be to consider coloring the first face red or green, then the second red or green and so on until we have a list of all the possibilities. This ignores an important fact about the problem: a cube looks the same from many points of view; i.e., it has symmetries. For example, there is only one way to color a cube so that it has one red face and five green ones. □

In the next four chapters, we'll study some fundamental techniques for counting and listing structures. What do we mean by "structures?" It is simply a general term for whatever things we are counting. We chose it rather than "thing" or "object" to emphasize that what we are counting has some internal organization. If the things we were trying to list or count did not have some sort of internal organization, we would have no way to systematically analyze them. A list of 30 distinct birthdays or a cube with colored faces are two examples of structures. Such a list has an internal organization: For the birthdays, we have 30 distinct days of the year written in some order. A cube has considerable structure because it can be rotated in various ways and end up occupying the same space.

Understanding the internal organization clearly is the first step in solving a counting or listing problem. For the birthdays, the organization is a sequence of 30 distinct numbers between 1 and 365 inclusive. For the cube, the organization is somehow tied to the cube's symmetries. We'll easily see how to answer the birthday question thanks to the simple description of the structure's organization. We have problems with the cube because we haven't yet come up with a clear description.

In Chapter 1 we'll study some simple structures—ordered and unordered lists with and without repetitions—and we'll introduce tools for counting them. The main tools are the Rules of Sum and Product. Recursions and generating functions will also appear briefly. We'll return to generating functions in Part IV.

In Chapter 2 we'll study functions and "permutations." Besides being of interest in themselves, functions provide another way to look at the material in Chapter 1, and permutations are essential for Chapter 4. We conclude Chapter 2 with a discussion of Boolean functions and combinatorial logic.

It is frequently necessary to generate combinatorial objects or accumulate information about them rather than just count them. In Chapter 3 you'll see how to use the function viewpoint and "trees" to generate lists of structures. Furthermore, we'll study how to access particular items in the list using "ranking." This is what we need for the biologist's problem. Trees are an important structure in computer science, so you'll encounter them again in this book.

In Chapter 4 we'll study two rather unrelated topics that did not merit a separate chapter. The first topic is counting and listing structures with symmetries. Our earlier notion of permutation provides the foundation for this discussion. The second topic is a method of counting structures in a somewhat indirect manner, called the "Principle of Inclusion and Exclusion."

Preliminary Reading

At various points in our discussion we will need to make use of proof by induction. In fact, induction is a more common proof technique in combinatorics than in most other branches of mathematics. We recommend that you review proof by induction in Appendix A (p. 361).

At times we will estimate values using "Big-Oh" and "little-oh" notation as well as the notation $f(n) \sim g(n)$. These are discussed in Section B.1 (p. 368). You may wish to look at this section quickly now and refer back to it as needed.

Since probability is a natural adjunct to counting, we'll encounter it from time to time in the examples and homework. The necessary background is reviewed in Appendix C (p. 381). You should look this over now and refer back to it as needed.

The algebraic rules for operating with sets are also familiar to most beginning university students. Here is such a list of the basic rules. In each case the standard name of the rule is given first, followed by the rule as applied first to $\cap$ and then to $\cup$.

Theorem 0.1 Algebraic rules for sets *The universal set U is not mentioned explicitly but is implicit when we use the notation $\sim X = U - X$ for the complement of X. An alternative notation is $X^c = \sim X$.*

Associative:	$(P \cap Q) \cap R = P \cap (Q \cap R)$	$(P \cup Q) \cup R = P \cup (Q \cup R)$
Distributive:	$P \cap (Q \cup R) = (P \cap Q) \cup (P \cap R)$	$P \cup (Q \cap R) = (P \cup Q) \cap (P \cup R)$
Idempotent:	$P \cap P = P$	$P \cup P = P$
Double Negation:	$\sim\sim P = P$	
DeMorgan:	$\sim(P \cap Q) = \sim P \cup \sim Q$	$\sim(P \cup Q) = \sim P \cap \sim Q$
Absorption:	$P \cup (P \cap Q) = P$	$P \cap (P \cup Q) = P$
Commutative:	$P \cap Q = Q \cap P$	$P \cup Q = Q \cup P$