Chapter 5 & 7
Counting

MAD101

Ly Anh Duong

duongla3@fe.edu.vn

Table of Contents

Example
1 The basics of Counting

- How many 6-letter passwords ****** are there (the letter can be choose from 0..9;a..z;A..Z)?
- How many bit strings of length n that do not have two consecutive 0s?
- ......

The product rule

1 The basics of Counting

How many ways to choose the cloths?

The product rule

Task=task 1 $\rightarrow$ task 2 $\rightarrow$...$\rightarrow$ task k

- Task 1: n_1 ways,
- Task 2: n_2 ways,
- ....
- Task k: n_k way.

Product rule: $n_1.n_2...n_k$ ways to do the task.

Example.

From city A to city B there are 2 paths, from B to C there are 3 paths.

a. How many ways are there from A through B to C?

b. If two more paths are opened from A to C (without going through B), how many ways are there from A to C?

Example.

From city A to city B there are 2 paths, from B to C there are 3 paths.

a. How many ways are there from A through B to C?

b. If two more paths are opened from A to C (without going through B), how many ways are there from A to C?

Ans: a.6;b.8

The sum rule

Task= task 1 or task 2 or....or task k

Task 1: n_1 ways,

Task 2: n_2 ways,

....

Task k: n_k way.

Sum rule: $n_1 + n_2 + ...n_k$ ways to do task.

Example. There are 37 faculties, 83 students. A person either a faculty or student can be choose to attend a committee. How many ways are there to select such person?

- First task (facalty): 37 ways
- Second task (student): 83 ways

Thus, Sum rule $\rightarrow 37 + 83 = 120$ ways.

Example

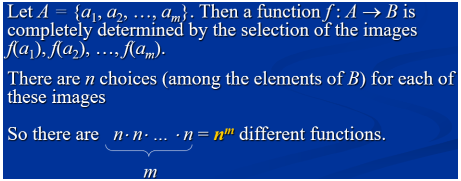
1. How many functions are there from an 3-element set A to an 4-element set B?
2. How many one-to-one functions are there from an 3-element set A to an 5-element set B?

Example

1. How many functions are there from an m-element set A to an n-element set B?
2. How many one-to-one functions are there from an m-element set A to an n-element set B?

Solution

1. .

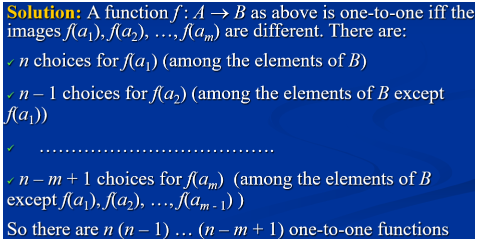


Let $A = \{a_1, a_2, \ldots, a_m\}$. Then a function $f : A \to B$ is completely determined by the selection of the images $f(a_1), f(a_2), \ldots, f(a_m)$.

There are n choices (among the elements of B) for each of these images

So there are $\underbrace{n \cdot n \cdot \ldots \cdot n}_{m} = n^m$ different functions.

2. .

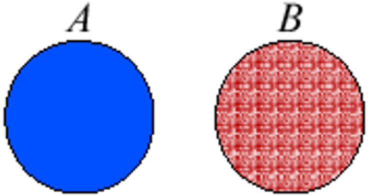


Solution: A function $f : A \to B$ as above is one-to-one iff the images $f(a_1), f(a_2), \ldots, f(a_m)$ are different. There are:

- ✓ n choices for $f(a_1)$ (among the elements of B)
- ✓ $n - 1$ choices for $f(a_2)$ (among the elements of B except $f(a_1)$)
- ✓ ……………………………….
- ✓ $n - m + 1$ choices for $f(a_m)$ (among the elements of B except $f(a_1), f(a_2), \ldots, f(a_{m-1})$)

So there are $n(n - 1) \ldots (n - m + 1)$ one-to-one functions

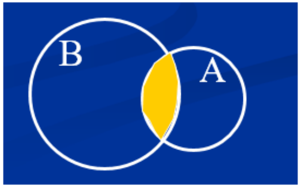
Phrased in terms of Set Theory



The sum and product rules can also be phrased in terms of Set Theory

- Sum rule $|A \cup B| = |A| + |B|$
- Product rule $|AB| = |A|.|B|$

The Inclusion-Exclusion Principle



$$|A \cup B| = |A| + |B| - |A \cap B|$$

Example. How many 8-bit strings either start with 1 or end with 00?

- Start with 1: 2^7
- End with 00: 2^6
- Start with 1 **and** end with 00: 2^5

Thus, we have $2^7 + 2^6 - 2^5 = 160$ 8-bit strings.

Table of Contents

Recurrence relations (hệ thức đệ quy)

- A **recurrence relation** for the sequence $\{a_n\}$ is an equations that express a_n in terms of the previous terms $a_0, a_1, \ldots, a_{n-1}$.
- The sequence $\{a_n\}$ is called a **solution** of the recurrence relation.

Example. Determine whether $a_n = 3n$ is a solution of $a_n = 2a_{n-1} - a_{n-2}$
We have $2a_{n-1} - a_{n-2} = 2.3(n-1) - 3(n-2) = 3n = a_n$

Example

2 Recurrence relations

Suppose $a_n = 2a_{n-1} - a_{n-2}$

- $a_n = 5$ is a solution because of $2a_{n-1} - a_{n-2} = 2.5 - 5 = 5 = a_n$
- $a_n = 2^n$ is not a solution because of $2a_{n-1} - a_{n-2} = 2.2^{n-1} - 2^{n-2} = 2^n - 2^{n-2} \neq a_n$

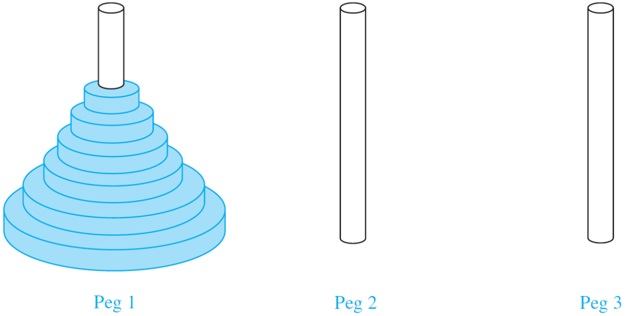
Note. A recurrence relation may have more than one Solution

Example.

Find the solution of $a_n = 2a_{n-1} - a_{n-2}, a_0 = 3, a_1 = 5$. (In this case, $a_0 = 3, a_1 = 5$ are called **initial conditions**)

- $a_2 = 2a_1 - a_0 = 2.5 - 3 = 7 = 3 + 2.2$
- $a_3 = 2a_2 - a_1 = 2.7 - 5 = 9 = 3 + 2.3$
- $a_4 = 2a_3 - a_2 = 2.9 - 7 = 11 = 3 + 2.4$
- ....
- $a_n = 2a_{n-1} - a_{n-2} = 3 + 2.n$ is a solution.

Modeling with Recurrence Relations: The Tower of Hanoi Problem



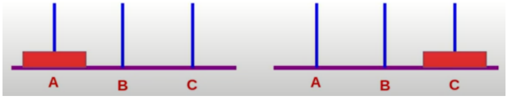
How can we move the disks to the 3^{nd} peg, one in a time, following the rule: **larger disks are never placed on top of smaller ones**?

Solution

2 Recurrence relations

Let H_n denote the number of moves needed to solve the Tower of Hanoi puzzle with n disks. Set up a recurrence relation for the sequence $\{H_n\}$

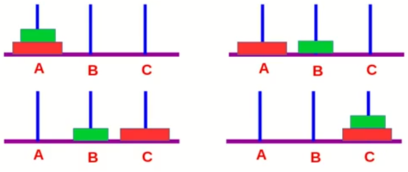
- $n = 1$



$H_1 = 1$

Solution

- $n = 2$



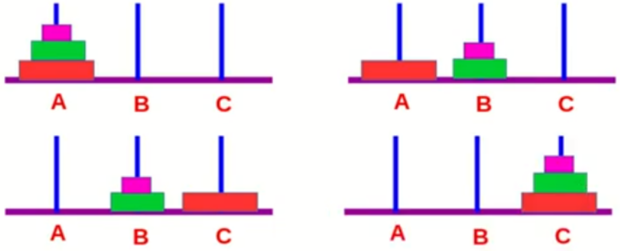
Move $1(2-1)$ disk from A to B.
Move 1 from A to C.
Move $1(2-1)$ from B to C.

Thus, $H_2 = 3$

Solution

2 Recurrence relations

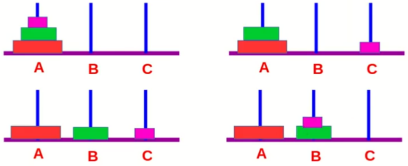
- $n = 3$



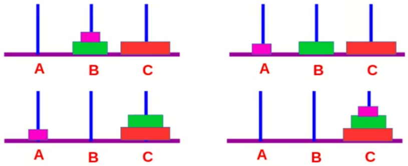
Move $2(3-1)$ disk from A to B.(?!)
Move 1 from A to C.
Move $2(3-1)$ from B to C.(?!)

Solution

Move $2(3-1)$ disk from A to B



Move $2(3-1)$ from B to C



Therefore, $H_3 = 7$

In General

We need to move n disks from A to C

Move $n-1$ disks from A to B.

Move 1 disk from A to C

Move $n-1$ disks from B to C

Thus, we have $H_1 = 1, H_n = 2H_{n-1} + 1$ is a recurrence relation.
To solve it, we write $H_n = 2H_{n-1} + 1 = 2[2H_{n-2} + 1] + 1 = 2^2 H_{n-2} + 2 + 1$
$= ... = 2^{n-1} H_1 + 2^{n-2} + ... + 2 + 1$
$H_1 = 1 \implies H_n = 2^n - 1$
Demo: https://mathsisfun.com/games/towerofhanoi.html
Note: $\sum_{k=0}^{n} ar^k = a\frac{1-r^{n+1}}{1-r} (r \neq 1).$

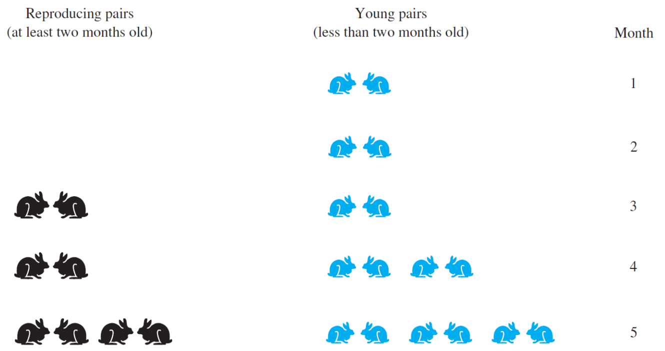
Algorithm

2 Recurrence relations

```
**Procedure** hanoi($n$,p1,p2,p3)
If $n = 1$ then
    Print ($p1$,"→",$p2$)
else
    hanoi($n-1, p1, p3, p2$)
    hanoi($1, p1, p2, p3$)
    hanoi($n-1, p3, p2, p1$)
```

Modeling with Recurrence Relations: Fibonacci numbers

A young pair of rabbits (1 male + 1 fem) is placed on an island. A 2 month old – pair will produce another pair. How many pairs of rabbits after n months?



Solution.

The number of pairs after the first few months are: $1, 1, 2, 3, 5, \ldots$

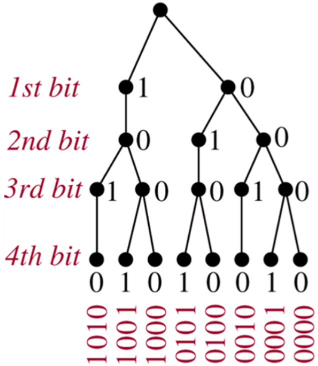
Let f_n be the number of pairs of rabbits after n months, the above results suggest the following relation:

$$f_n = f_{n-1} + f_{n-2}$$

- With the initial conditions $f_1 = 1$ and $f_2 = 1$ the solution is unique
- The numbers fn with the above initial conditions are also called the **Fibonacci numbers.**

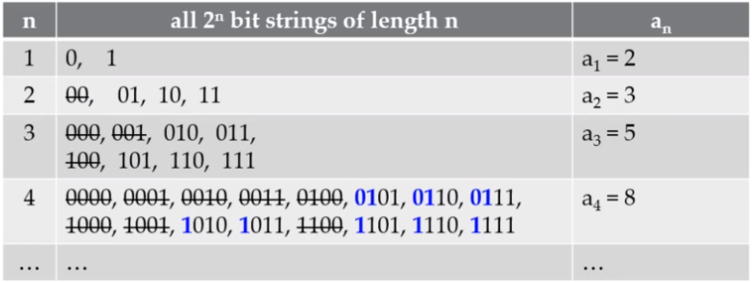
Use recurrence relations for counting

Find recurrence relation and give initial conditions for **the number of bit strings of length n that do not have two consecutive 1s**. How many such bit strings are there of length four.



Solution.

- n length of bit strings,
- Let a_n: number of bit string of length n that do not have two consecutive 0s.



n	all 2^n bit strings of length n	a_n
1	0, 1	$a_1 = 2$
2	~~00~~, 01, 10, 11	$a_2 = 3$
3	~~000~~, ~~001~~, 010, 011, ~~100~~, 101, 110, 111	$a_3 = 5$
4	~~0000~~, ~~0001~~, ~~0010~~, ~~0011~~, ~~0100~~, 0101, 0110, 0111, ~~1000~~, ~~1001~~, 1010, 1011, ~~1100~~, 1101, 1110, 1111	$a_4 = 8$
…	…	…

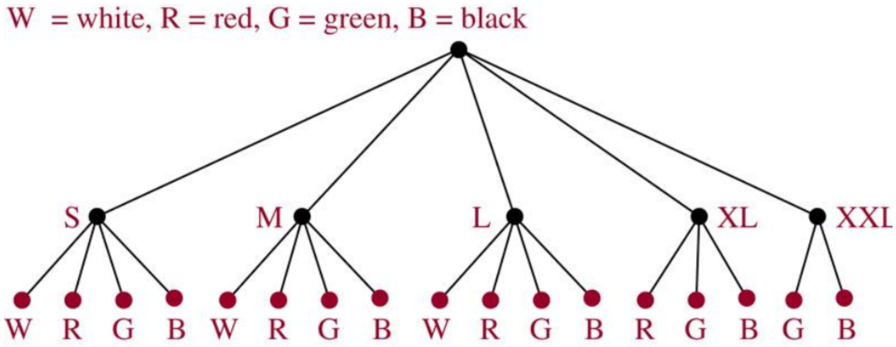
Tree diagram

Counting problems can be solved using tree diagrams.

Example. Suppose that “I Love New Jersey” T-shirts come in five different sizes: S, M, L, XL, and XXL. Further suppose that each size comes in four colors, white, red, green, and black, except for XL, which comes only in red, green, and black, and XXL, which comes only in green and black. How many different shirts does a souvenir shop have to stock to have at least one of each available size and color of the T-shirt?

Solution

2 Recurrence relations


W = white, R = red, G = green, B = black
S
M
L
XL
XXL
W R G B W R G B W R G B R G B G B

Thus, the souvenir shop owner needs to stock 17 different T-shirts.

Modeling with Recurrence Relations: Compound interest Problem (lãi kép)

A sum of P_0 is deposited in the saving account with the interest rate of $r\%$ compounded annually. How much will be in the account after n years?

Solution. Let P_n be the amount after n years, then P_n is obtained from P_{n-1} and the interest yielded from that amount in one year:

$$P_n = P_{n-1} + rP_{n-1} = (1+r)P_{n-1} = (1+r)^2 P_{n-2} = ... = (1+r)^n P_0$$

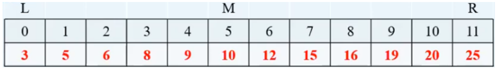
Thus, $P_n = (1+r)^n P_0$

Table of Contents

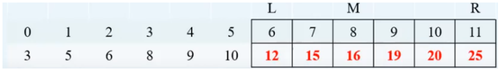
3 Divide & Conquer Algorithms

Divide-and-Conquer Algorithms

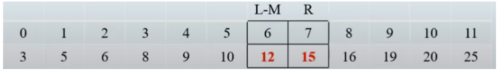
Example. 3, 5, 6, 8, 9, 11, 12, 15, 16, 19, 20, 25. Use Binary Search, find 12.



L					M						R
0	1	2	3	4	5	6	7	8	9	10	11
3	5	6	8	9	10	12	15	16	19	20	25



						L		M			R
0	1	2	3	4	5	6	7	8	9	10	11
3	5	6	8	9	10	12	15	16	19	20	25



						L-M	R				
0	1	2	3	4	5	6	7	8	9	10	11
3	5	6	8	9	10	12	15	16	19	20	25

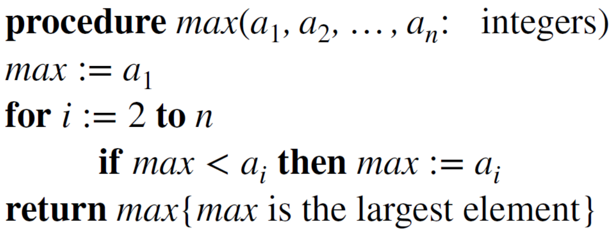
This is a Divide-and-Conquer Algorithms.

Divide-and-Conquer Algorithms

- To solve a problem of size n,
- (Divide.) We divide the original problem into subproblems that are similar to the original problem but of size smaller than n,
- (Conquer.) Solving the subproblems and summarizing the solutions, we get the solution of the original problem,
- For subproblems we also apply the same approach,
- This division gives us basic problems that are easy to solve,
- $\to$ Recurrence relations algorithms.

Example

Remind the algorithm



```
procedure $max(a_1, a_2, \ldots, a_n$: integers)
$max := a_1$
for $i := 2$ to $n$
        if $max < a_i$ then $max := a_i$
return $max$ {$max$ is the largest element}
```

The number of comparisons is $f(n) = 2n - 1$. So, $f(n)$ is $O(n)$

Divide and conquer

$$\frac{\text{max}(n \text{ int}) \qquad \text{f(n)}}{\text{max}(n/2 \text{ int}) \qquad \text{f(n/2)} \qquad \text{max}(n/2 \text{ int})}$$

$\rightarrow f(n) = 2f(n/2) + 1$ How about the complexity ?

Example.

3 Divide & Conquer Algorithms

```
**procedure** *linear search*($x$: integer, $a_1, a_2, \ldots, a_n$: distinct integers)
$i := 1$
**while** ($i \le n$ and $x \ne a_i$)
        $i := i + 1$
**if** $i \le n$ **then** $location := i$
**else** $location := 0$
**return** $location$ {$location$ is the subscript of the term that equals $x$, or is 0 if $x$ is not found}
```

Example.

```
**procedure** *linear search*($x$: integer, $a_1, a_2, \ldots, a_n$: distinct integers)
$i := 1$
**while** ($i \le n$ and $x \ne a_i$)
    $i := i + 1$
**if** $i \le n$ **then** $location := i$
**else** $location := 0$
**return** $location$ {$location$ is the subscript of the term that equals $x$, or is 0 if $x$ is not found}
```

$f(n) = 2n + 1$ is $O(n)$

Example.

```
procedure binary search($i, j, x$: integers, $1 \le i \le j \le n$)
$m := \lfloor (i + j)/2 \rfloor$
if $x = a_m$ then
    return $m$
else if ($x < a_m$ and $i < m$) then
    return binary search($i, m - 1, x$)
else if ($x > a_m$ and $j > m$) then
    return binary search($m + 1, j, x$)
else return 0
{output is location of $x$ in $a_1, a_2, \ldots, a_n$ if it appears; otherwise it is 0}
```

$f(n) = 1.f(n/2) + 2$ How about the complexity ?

Divide-and-conquer recurrence relation

- n: size of the original problem,
- a: numbers of sub-problems must be conquered,
- n/b : size of the sub-problem,
- $f(n)$: number of operation required to solve the original problem,
- $\rightarrow f(n/b)$: number of operation required to solve a sub-problem,
- $g(n)$: overhead for additional work of the step conquer.

Divide-and-conquer recurrence relation:

$$\boxed{f(n) = af\left(\frac{n}{b}\right) + g(n)}$$

Example.

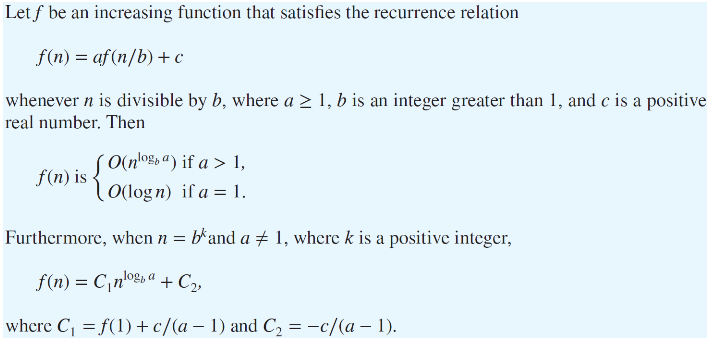
In binary search, suppose n even. We have

- $a = 1$
- $b = 2$
- $g(n) = 2$

Thus, $f(n) = f\left(\frac{n}{2}\right) + 2$

Theorem

3 Divide & Conquer Algorithms



Let f be an increasing function that satisfies the recurrence relation

$$f(n) = af(n/b) + c$$

whenever n is divisible by b, where $a \geq 1$, b is an integer greater than 1, and c is a positive real number. Then

$$f(n) \text{ is } \begin{cases} O(n^{\log_b a}) & \text{if } a > 1, \\ O(\log n) & \text{if } a = 1. \end{cases}$$

Furthermore, when $n = b^k$ and $a \neq 1$, where k is a positive integer,

$$f(n) = C_1 n^{\log_b a} + C_2,$$

where $C_1 = f(1) + c/(a-1)$ and $C_2 = -c/(a-1)$.

Note. $n = b^k \implies n^{log_b a} = (b^k)^{log_b a} = b^{log_b a^k} = a^k$. Hence,

$$\boxed{f(n) = C_1 a^k + C_2 = a^k[f(1) + c/(a-1)] - c/(a-1)}$$

Example.

1. $f(n) = 5f(n/2) + 3, f(1) = 7$. Find $f(2^k), k$ is a positive integer. Estimate $f(n)$ if f is increasing function.
2. Estimate the number of comparisons used by a binary search $f(n) = f(n/2) + 2$
3. Estimate the number of comparisons to locate the maximum element in a sequence $f(n) = 2f(n/2) + 1$

1. $f(2^k) = 5^k[f(1)+c/(a-1)]-c/(a-1) = 5^k[7 + 3/4] - 3/4 = \frac{31}{4}.5^k - \frac{3}{4}$
   $a = 5 > 1$, f increasing, so $f(n) = O(n^{log5})$
2. $a = 1 \implies f(n) = O(logn)$
3. $a = 2 > 1 \implies f(n) = O(n^{log2}) = O(n)$

Master Theorem

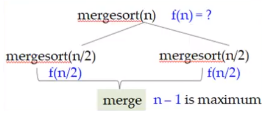
3 Divide & Conquer Algorithms

MASTER THEOREM Let f be an increasing function that satisfies the recurrence relation

$$f(n) = af(n/b) + cn^d$$

whenever $n = b^k$, where k is a positive integer, $a \geq 1$, b is an integer greater than 1, and c and d are real numbers with c positive and d nonnegative. Then

$$f(n) \text{ is } \begin{cases} O(n^d) & \text{if } a < b^d, \\ O(n^d \log n) & \text{if } a = b^d, \\ O(n^{\log_b a}) & \text{if } a > b^d. \end{cases}$$



- $f(n) = 2f(n/2) + n \rightarrow$
  $a = 2, b = 2, d = 1, c = 1 \rightarrow a = b^d$
- $\rightarrow f(n)$ is $O(nlogn)$

Example

1. Complexity of Merge Sort $M(n) = 2M(\frac{n}{2}) + n$.
2. Give a big-O estimate for the number of bit operations needed to multiply two n-bit integers using the fast multiplication algorithm $f(n) = 3f(\frac{n}{2}) + Cn$.
3. Give a big-O estimate for the number of multiplications and additions required to multiply two nxn matrices using the matrix multiplication algorithm $f(n) = 7f(\frac{n}{2}) + 15\frac{n^2}{4}$, when n is even.

Solution

1. $a = 2 = 2^1 = b^d \implies M(n) = O(nlogn)$.
2. $a = 3 > 2^1 = b^d \implies f(n) = O(n^{log3})$
3. $a = 7 > 2^2 = b^d \implies f(n) = O(n^{log7})$

Table of Contents

The Basics of Counting

1. How many different bit strings of length seven are there?

2. How many different license plates can be made if each plate contains a sequence of three uppercase English letters followed by three digits (and no sequences of letters are prohibited, even if they are obscene)?

3. **Counting Functions**: How many functions are there from a set with m elements to a set with n elements?

4. **Counting One-to-One Functions**: How many one-to-one functions are there from a set with m elements to one with n elements?

5. **Counting bijection Functions**: How many one-to-one functions are there from a set with m elements to one with n elements?

The Basics of Counting

6. Each user on a computer system has a password, which is six to eight characters long, where each character is an uppercase letter or a digit. Each password must contain at least one digit. How many possible passwords are there?

7. How many bit strings of length eight either start with a 1 bit or end with the two bits 00?

8. How many bit strings of length four do not have two consecutive 1s?

9. How many bit strings of length ten both begin and end with a 1?

10. How many positive integers between 5 and 31

a) are divisible by 3? b) are divisible by 4? c) are divisible by 3 and by 4?

The Basics of Counting

11. How many positive integers between 50 and 100

a) are divisible by 7? b) are divisible by 11? c) are divisible by both 7 and 11?

12. How many positive integers less than 1000

a) are divisible by 7?

b) are divisible by 7 but not by 11?

c) are divisible by both 7 and 11?

d) are divisible by either 7 or 11?

e) are divisible by exactly one of 7 and 11?

f) are divisible by neither 7 nor 11?

g) have distinct digits?

h) have distinct digits and are even?

The Basics of Counting

13. How many positive integers between 100 and 999 inclusive

a) are divisible by 7?

b) are odd?

c) have the same three decimal digits?

d) are not divisible by 4?

e) are divisible by 3 or 4?

f) are not divisible by either 3 or 4?

g) are divisible by 3 but not by 4?

h) are divisible by 3 and 4?

14. How many one-to-one functions are there from a set with five elements to sets with the following number of elements?

a) 4 b) 5 c) 6 d) 7

15. How many bit strings of length seven either begin with two 0s or end with three 1s?

16. How many bit strings of length 10 either begin with three 0s or end with two 0s?

17. How many bit strings of length 8 begin with 11 or end with 00?

18. Let B be the set {a, b, c}. How many functions are there from B^2 to B?

19. How may bit strings with length 10 that has exactly three 1s and end with 0?

20. A game consisting of flipping a coin ends when the player gets two heads in a row, two tails in a row or flips the coin four times. In how many ways can the game end?

Recurrence Relations

1. a) Find a recurrence relation for the number of permutations of a set with n elements.

b) Use this recurrence relation to find the number of permutations of a set with n elements using iteration.

2. a) Find a recurrence relation for the number of bit strings of length n that do not contain three consecutive 0s.

b) What are the initial conditions?

c) How many bit strings of length seven do not contain

three consecutive 0s?

3. What is the solution of the recurrence relation $a_n = a_{n-1} + 2a_{n-2}$ with $a_0 = 2$ and $a_1 = 7$?

Divide-and-Conquer Algorithms and recurrence Relations

1. How many comparisons are needed for a binary search in a set of 64 elements?

2. Suppose that f(n) = f(n/3) + 1 when n is a positive integer divisible by 3, and f(1) = 1. Find

a) f(3) b) f(27) c) f(729)

3. Suppose that f(n) = 2f(n/2) + 3 when n is an even positive integer, and f(1) = 5. Find

a) f(2) b) f(8) c) f(64) d) f(1024).

4. Suppose that f(n) = f(n/5) + 3n when n is a positive integer divisible by 5, and f(1) = 4. Find

a) f(5) b) f(125) c) f(3125)

Q&A

Thank you for listening!