## Content

<table>
<thead>
<tr>
<th></th>
<th>Understanding and analyzing algorithms</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Using graphs and graph algorithms</td>
</tr>
<tr>
<td>3</td>
<td>Using combinatorial reasoning to quantitatively analyze algorithms and systems</td>
</tr>
<tr>
<td>4</td>
<td>Using probability to algorithm analysis</td>
</tr>
</tbody>
</table>

3.1 Basics of Counting: Product rule, Sum rule  
3.2 Permutations, Combinations  
3.3 Lower bound for comparison-based sorting  
3.4 Fixed density binary strings
<table>
<thead>
<tr>
<th></th>
<th>Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Understan... and analyzing algorithms</td>
</tr>
<tr>
<td>2</td>
<td>Using graphs and graph algorithms</td>
</tr>
<tr>
<td>3</td>
<td>Using combinatorial reasoning to quantitatively analyze algorithms and systems</td>
</tr>
<tr>
<td>4</td>
<td>Using probability to algorithm analysis</td>
</tr>
</tbody>
</table>

3.1 Basics of Counting: Product rule, Sum rule
3.2 Permutations, Combinations
3.3 Lower bound for comparison-based sorting
3.4 Fixed density binary strings
Basics of Counting: Product Rule and Sum Rule
What do we mean by counting?

• How many arrangements or combinations of objects are there of a given form?
• How many of these have a certain property?

Math = “search for order”
Counting is important for Computer Scientists

- **Hardware**: How many ways are there to arrange components on a chip?
- **Algorithms**: How long is this loop going to take? How many times does it run?
- **Security**: How many passwords are there?
- **Memory**: How many bits of memory should be allocated to store an object?
Memory requirements

How many bits of memory should be allocated to store a decimal numbers with $n$ digits?
Memory requirements

How many bits of memory should be allocated to store a decimal numbers with $n$ digits?

<table>
<thead>
<tr>
<th>$n$</th>
<th>#Bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>9 $(1001)_2$</td>
</tr>
<tr>
<td>2</td>
<td>99 $(110 0011)_2$</td>
</tr>
<tr>
<td>3</td>
<td>999 $(11 1110 0111)_2$</td>
</tr>
<tr>
<td>4</td>
<td>9999 $(10 0111 0000 1111)_2$</td>
</tr>
</tbody>
</table>

$$\lceil \log_2(10^n) \rceil = \lceil n \cdot \log_2(10) \rceil \approx \lceil 3.32n \rceil$$

Moral:
If we care about storage requirements, we must care about counting.
Product Rule (1/3)

• Say you are designing a video game where each player can choose a preset character or make their own character with custom facial features.

• How many different characters can be created?

• Say you can only choose from these 12 hair styles, and these 8 hair colors.

• How many different characters can you create?
Product Rule (2/3)

- Say you are designing a video game where each player can choose a preset character or make their own character with custom facial features.
- How many different characters can be created?

- Say you can only choose from these 12 hair styles, and these 8 hair colors.
- How many different characters can you create?

Answer: $12 \times 8 = 96$
Product Rule: For any sets $A$ and $B$, $|A \times B| = |A| \times |B|$.

- In our example,
  
  $A = \{$hair styles$}, \ |A| = 12$
  
  $B = \{$hair colors$}, \ |B| = 8$

- Specifying a character means giving an ordered pair (hair style, hair color).

- The Product Rule says the number of such pairs is $12 \times 8 = 96$. 
Sum Rule (1/3)

- Say you can choose one of the 96 custom characters or use a preset character.
- If there are 10 preset characters you can choose from, how many different characters can you be?
• Say you can choose one of the 96 custom characters or use a preset character.
• If there are 10 preset characters you can choose from, how many different characters can you be?

Answer: $96 + 10 = 106$
Sum Rule (3/3)

Sum Rule: For any disjoint sets A and B,

$$|A \cup B| = |A| + |B|.$$ 

- In our example,
  
  $$A = \{\text{custom characters}\}, \ |A| = 96$$
  $$B = \{\text{preset characters}\}, \ |B| = 10$$

- These sets are disjoint since there is no overlap.
- The Sum Rule says the total number of characters is
  $$96 + 10 = 106.$$
Sum Rule: For any *disjoint* sets $A$ and $B$, 

$$|A \cup B| = |A| + |B|.$$  

- Disjointness is necessary, otherwise the formula is not true.
Sum Rule: Inclusion/Exclusion for two sets

Inclusion-Exclusion: For any sets A and B,

\[ |A \cup B| = |A| + |B| - |A \cap B| \]

# people who know Java OR C
= # people who know Java
+ # people who know C
- # people who know Java AND C

People who know Java
People who know C
Sum Rule: Inclusion/Exclusion for three sets

\[ |A \cup B \cup C| = |A| + |B| + |C| - |A \cap B| - |B \cap C| - |C \cap A| + |A \cap B \cap C| \]
### Example Questions

**Favorite Sports:**

<table>
<thead>
<tr>
<th></th>
<th>Football</th>
<th>Baseball</th>
<th>Soccer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Brian</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Carl</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Christine</td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Jianhan</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Louise</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Ron</td>
<td></td>
<td>X</td>
<td></td>
</tr>
</tbody>
</table>

Illustrate the Sum Rule together with Inclusion/Exclusion for the given table.
Example Questions

Favorite Sports:

<table>
<thead>
<tr>
<th></th>
<th>Football</th>
<th>Baseball</th>
<th>Soccer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Brian</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Carl</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Christine</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Jianhan</td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Louise</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Ron</td>
<td></td>
<td>X</td>
<td></td>
</tr>
</tbody>
</table>

Illustrate the Sum Rule together with Inclusion/Exclusion for the given table.

Solution: There are 3 (Football), 4 (Basketball), 4 (Soccer) X, so in total 11 X. But there are only 6 people. The computation goes as follows:

11 – (Football and Baseball) – (Football and Soccer) – (Soccer and Baseball) + (Football and Baseball and Soccer) =

11 – 2 -2 -2 + 1 = 6
Example Questions

The chairs of an auditorium are to be labeled with one of the 26 uppercase English letters followed by a positive integer not exceeding 100. What is the largest number of chairs that can be labeled differently?
Example Questions

The chairs of an auditorium are to labeled with one of the 26 uppercase English letters followed by a positive integer not exceeding 100. What is the largest number of chairs that can be labeled differently?

Solution:

26 \times 100 = 2,600 \text{ different ways that a chair can be labeled.}
Example Questions

- **Example:** At an ice cream parlor, you can have your ice cream in a bowl, cake cone, or sugar cone. There are 20 different flavors available. How many single-scoop creations?
Example Questions

• Example: At an ice cream parlor, you can have your ice cream in a bowl, cake cone, or sugar cone. There are 20 different flavors available. How many single-scoop creations?

• Solution:
  - 3 choices for your vehicle
  - 20 choices for your flavor (independent of the vehicle you chose)
  - So 3 * 20 = 60 creations
Example Questions

- **Example**: At a different ice cream parlor, the only flavors are vanilla, chocolate, and strawberry. You can order a waffle cone or a sundae. Sundaes come with your choice of caramel or hot fudge. Whipped cream and a cherry are optional.

- Note that the number of toppings depends on whether you choose a cone or a sundae.

- Break into disjoint cases:
  - 1) Cone [3 choices]
  - 2) Sundae [s choices]

- Number of desserts = 3 + s
Example Questions

- **Example:** At a different ice cream parlor, the only flavors are vanilla, chocolate, and strawberry. You can order a waffle cone or a sundae. Sundaes come with your choice of caramel or hot fudge. Whipped cream and a cherry are optional.

- Note that the number of toppings depends on whether you choose a cone or a sundae.

- Break into disjoint cases:
  - 1) Cone [3 choices]
  - 2) Sundae [s choices]

- Number of desserts = 3 + s

- A sundae is a sequence or a tuple:
  (vanilla/chocolate/strawberry, caramel/hot fudge, whipped cream/none, cherry/none)

- Choices in each category are completely independent

- Number of sundaes:
  \[ s = 3 \cdot 2 \cdot 2 \cdot 2 = 24 \]
Example Questions

- **Example:** At a different ice cream parlor, the only flavors are vanilla, chocolate, and strawberry. You can order a waffle cone or a sundae. Sundaes come with your choice of caramel or hot fudge. Whipped cream and a cherry are optional.

- Note that the number of toppings *depends on* whether you choose a cone or a sundae.

- Break into disjoint cases:
  - 1) Cone [3 choices]
  - 2) Sundae [s choices]

- Number of desserts = 3 + s

- A sundae is a sequence or a tuple: (vanilla/chocolate/strawberry, caramel/hot fudge, whipped cream/none, cherry/none)

- Choices in each category are completely independent

- Number of sundaes:

  \[ s = 3 \times 2 \times 2 \times 2 = 24 \]

- In total there are

  \[ 3 + s = 3 + 24 = 27 \]

dessert options.
Example Questions

Each user in 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?
Example Questions

Each user in 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?

Solution:
Let $P$ be the total number of possible passwords, and let $P_6$, $P_7$, and $P_8$ denote the number of possible passwords of length 6, 7, and 8, respectively. $P = P_6 + P_7 + P_8$. 
Example Questions

Each user in 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?

Solution:
Let P be the total number of possible passwords, and let P₆, P₇, and P₈ denote the number of possible passwords of length 6, 7, and 8, respectively. 
P=P₆+P₇+P₈.

Number of strings of six characters:
Number of strings with no digits:
Example Questions

Each user in 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?

Solution:
Let \( P \) be the total number of possible passwords, and let \( P_6, P_7, \) and \( P_8 \) denote the number of possible passwords of length 6, 7, and 8, respectively.
\[
P = P_6 + P_7 + P_8.
\]

Number of strings of six characters: \((26+10)^6\)
Number of strings with no digits: \(26^6\)
Example Questions

Each user in 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?

Solution:
Let \( P \) be the total number of possible passwords, and let \( P_6, P_7, \) and \( P_8 \) denote the number of possible passwords of length 6, 7, and 8, respectively.
\[
P = P_6 + P_7 + P_8.
\]

Number of strings of six characters: \((26+10)^6\)
Number of strings with no digits: \(26^6\)

Hence, \( P_6 = 36^6 - 26^6 = 1.867.866.560 \)

Similarly,
\[
P_7 = 36^7 - 26^7 = 70.332.353.920
\]
and
\[
P_8 = 36^8 - 26^8 = 2.612.282.842.880
\]

Consequently, \( P = P_6 + P_7 + P_8 = 2.684.483.063.360. \)
Example Questions

A computer company receives 350 applications from graduates. 220 of these applicants majored in computer science, 147 in business, and 51 majored both in computer science and in business. How many of the applicants majored neither in computer science nor in business?
Example Questions

A computer company receives 350 applications from graduates. 220 of these applicants majored in computer science, 147 in business, and 51 majored both in computer science and in business. How many of the applicants majored neither in computer science nor in business?

Solution:

\[ 350 - (220 + 147 - 51) = 350 - 316 = 34 \]

34 of the applicants majored neither in computer science nor in business.
<table>
<thead>
<tr>
<th></th>
<th>Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Understanding and analyzing algorithms</td>
</tr>
<tr>
<td>2</td>
<td>Using graphs and graph algorithms</td>
</tr>
<tr>
<td>3</td>
<td>Using combinatorial reasoning to quantitatively analyze algorithms and systems</td>
</tr>
<tr>
<td>4</td>
<td>Using probability to algorithm analysis</td>
</tr>
</tbody>
</table>

3.1 Basics of Counting: Product rule, Sum rule
3.2 Permutations, Combinations
3.3 Lower bound for comparison-based sorting
3.4 Fixed density binary strings
Permutations

Combinations
Binomial Coefficients and Identities

Let $n$ and $k$ be nonnegative integers with $k \leq n$. Then

1. $C(n, k) = \binom{n}{k} = \frac{n!}{k!(n-k)!}$ (Binomial Coefficient)

2. $\sum_{k=0}^{n} \binom{n}{k} = 2^n$

3. $\sum_{k=0}^{n} (-1)^k \binom{n}{k} = 0$

4. $\sum_{k=0}^{n} 2^k \binom{n}{k} = 3^n$

5. $\binom{n+1}{k} = \binom{n}{k-1} \binom{n}{k}$ (Pascal’s Identity)

6. $\sum_{k=0}^{n} \binom{n}{k}^2 = \binom{2n}{n}$
### Permutations and Combinations

<table>
<thead>
<tr>
<th>No Repetition of Objects</th>
<th>When Some of the Objects Are Not Distinguishable</th>
<th>With Repetition of Objects</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Permutation</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ordered arrangement of ( k ) out of ( n ) objects</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Combination</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Unordered selection of ( k ) out of ( n ) objects</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Permutations**

- **A 100-Sprint**
  \[
  \frac{n!}{(n-k)!}
  \]

- **B Necklace**
  \[
  \frac{n!}{n! \cdot \ldots \cdot n_k!}
  \]

- **C Yes/No/Maybe**
  \[
  n^k
  \]

**Combinations**

- **D Lottery**
  \[
  \binom{n}{k}
  \]

- **E Dice**
  \[
  \binom{n+k-1}{k}
  \]
### Permutation

**Ordered arrangement of** \( k \) **out of** \( n \) **objects**

### A 100m-Sprint

\( n = 3 \) sprinters A, B, C.
Possible outcomes for the first \( k = 2 \) places (no ties).

**Objects:**
Sprinters
A, B, C (first place)
A, B, C (second place)

\[
\begin{array}{c|c|c}
\text{AB} & \text{BA} & \text{CA} \\
\text{AC} & \text{BC} & \text{--} \\
\end{array}
\]

### B Necklace

\( n = 3 \) beads red, red, green.
Possible different patterns when you thread \( n_1 = 2 \) red, \( n_2 = 1 \) green beads on a necklace.

**Objects:**
Beads
r, r, g (first position)
r, r, g (second position)
r, r, g (third position)

\[
\begin{array}{c|c|c|c}
\text{rbg} & \text{brg} & \text{grb} \\
\text{rgb} & \text{bgr} & \text{grr} \\
\text{rrg} & \text{rgr} & \text{grr} \\
\end{array}
\]

### C Yes/No/Maybe

\( n = 3 \) answers yes, no, maybe.
Possible outcomes for \( k = 2 \) questions.

**Objects:**
Answers
y, n, m (first question)
y, n, m (second question)

\[
\begin{array}{c|c|c|c}
\text{yy} & \text{ny} & \text{my} \\
\text{yn} & \text{nn} & \text{mn} \\
\text{ym} & \text{nm} & \text{mm} \\
\end{array}
\]
Combination

Unordered selection of \( k \) out of \( n \) objects

\[
\binom{n}{k}
\]

D Lottery

\( n = 5 \) numbers 1, 2, 3, 4, 5.
Possible outcomes for drawing \( k = 2 \) out of the \( n = 5 \) numbers.

Objects:
Numbers
1, 2, 3, 4, 5 (first number)
1, 2, 3, 4, 5 (second number)

--
12 --
13 23 --
14 24 34 --
15 25 35 45 --

E Dice

\( n = 6 \) numbers 1, 2, 3, 4, 5, 6.
Possible outcomes for \( k = 2 \) dice.

Objects:
Numbers
1, 2, 3, 4, 5, 6 (first die)
1, 2, 3, 4, 5, 6 (second die)

11
12 22
13 23 33
14 24 34 44
15 25 35 45 55
16 26 36 46 56 66

Combination

D Lottery

\( n = 5 \) numbers 1, 2, 3, 4, 5.
Possible outcomes for drawing \( k = 2 \) out of the \( n = 5 \) numbers.

Objects:
Numbers
1, 2, 3, 4, 5 (first number)
1, 2, 3, 4, 5 (second number)

--
12 --
13 23 --
14 24 34 --
15 25 35 45 --

E Dice

\( n = 6 \) numbers 1, 2, 3, 4, 5, 6.
Possible outcomes for \( k = 2 \) dice.

Objects:
Numbers
1, 2, 3, 4, 5, 6 (first die)
1, 2, 3, 4, 5, 6 (second die)

11
12 22
13 23 33
14 24 34 44
15 25 35 45 55
16 26 36 46 56 66
Example Questions

How many different 3-digit numbers can you form out of the digits 1, 2, 3?
Each digit can occur only once in a number.
Hint: the objects are 1, 2, 3
Example Questions

How many different 3-digit numbers can you form out of the digits 1, 2, 3? Each digit can occur only once in a number.
Hint: the objects are 1, 2, 3

Solution:
• Order is important \(\rightarrow\) permutation
• no repetition of objects
• all objects are distinguishable \(\rightarrow\) k-permutation (\(k=n=3\))
Example Questions

How many different 3-digit numbers can you form out of the digits 1, 2, 3?
Each digit can occur only once in a number.
Hint: the objects are 1, 2, 3

Solution:
- Order is important → permutation
- no repetition of objects
- all objects are distinguishable → k-permutation (k=n=3)

\[
\frac{n!}{(n - k)!}
\]

\(n=3\) sprinters A, B, C.
Possible outcomes for the first \(k=2\) places (no ties).

Objects:
Sprinters
A, B, C (first place)
A, B, C (second place)

-- BA CA
AB -- CB
AC BC --
Example Questions

How many different committees of three students can be formed from a group of four students?
Hint: the objects (students) are numbered 1, 2, 3, 4
Example Questions

How many different committees of three students can be formed from a group of four students?
Hint: the objects (students) are numbered 1, 2, 3, 4

Solution:
• Order doesn’t matter → combination
• no repetition of objects → k-combination (k=3, n=4)
Example Questions

How many different committees of three students can be formed from a group of four students?

Hint: the objects (students) are numbered 1, 2, 3, 4

Solution:

• Order doesn‘t matter → combination
• no repetition of objects → k-combination (k=3, n=4)

D Lottery
\[
\binom{n}{k}
\]

n=5 numbers 1, 2, 3, 4, 5.
Possible outcomes
for drawing k=2
out of the n=5 numbers.

Objects:
Numbers
1, 2, 3, 4, 5 (first number)
1, 2, 3, 4, 5 (second number)

123
124
134
234

<table>
<thead>
<tr>
<th></th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>23</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>24</td>
<td>34</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>25</td>
<td>35</td>
<td>45</td>
<td></td>
</tr>
</tbody>
</table>
Example Questions

Suppose a department contains 10 men and 15 women. How many ways are there to form a committee with six members if it must have the same number of men and women?
Example Questions

Suppose a department contains 10 men and 15 women. How many ways are there to form a committee with six members if it must have the same number of men and women?

Solution:

- We have to choose 3 out of 10 men and 3 out of 10 women.
- Order doesn’t matter → combination
- no repetition of objects → k-combination

men: k=3, n=10
women: k=3, n=15

\[ \binom{10}{3} \cdot \binom{15}{3} = \frac{10 \cdot 9 \cdot 8}{3 \cdot 2} \cdot \frac{15 \cdot 14 \cdot 13}{3 \cdot 2} = 120 \cdot 455 = 54,600 \]

D Lottery

\[ \binom{n}{k} \]

n=5 numbers 1, 2, 3, 4, 5. Possible outcomes for drawing k=2 out of the n=5 numbers.

Objects:
Numbers
1, 2, 3, 4, 5 (first number)
1, 2, 3, 4, 5 (second number)

D Lottery

\[ \binom{n}{k} \]

n=5 numbers 1, 2, 3, 4, 5. Possible outcomes for drawing k=2 out of the n=5 numbers.

Objects:
Numbers
1, 2, 3, 4, 5 (first number)
1, 2, 3, 4, 5 (second number)
More Examples

Homework and Rosen, Chapter 6.
### Content

<table>
<thead>
<tr>
<th></th>
<th>Understanding and analyzing algorithms</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Using graphs and graph algorithms</td>
</tr>
<tr>
<td>2</td>
<td>Using combinatorial reasoning to quantitatively analyze algorithms and systems</td>
</tr>
<tr>
<td>3</td>
<td>Using probability to algorithm analysis</td>
</tr>
</tbody>
</table>

3.1 Basics of Counting: Product rule, Sum rule
3.2 Permutations, Combinations
3.3 Lower bound for comparison-based sorting
3.4 Fixed density binary strings

---

1 Analyzing Algorithms | 2 Graphs and graph algorithms | 3 Combinatorial reasoning | 4 Probability
3.1 Counting | 3.2 Permutations/Combinations | 3.3 Lower bound Sorting | 3.4 Fixed Density Binary Strings
Lower bound for comparison-based sorting
Lower bound for comparison-based sorting

We measure the cost of a sorting algorithm in the number of comparisons between array elements.

We will show in this section that best comparison-based sorting algorithms are $\Theta(n\log(n))$

(like MergeSort)

So it is impossible to have a comparison-based algorithm that does better than this in the worst case.
We can construct a *branching diagram* or *decision tree* that shows the possible comparisons we might have to do.

**Binary Tree**

*Internal vertices:* Comparisons

*Leaves:* All possible sorted orders (permutations) of the array of size n

**Question 1:** How many leaves are there?
Decision Tree

We can construct a *branching diagram* or *decision tree* that shows the possible comparisons we might have to do.

**Binary Tree**

**Internal vertices:**
Comparisons

**Leaves:**
All possible sorted orders (permutations) of the array of size $n$

**Question 1:** How many leaves are there?

$n!$
We can construct a branching diagram or decision tree that shows the possible comparisons we might have to do.

![Decision Tree Diagram]

**Binary Tree**
- Internal vertices: Comparisons
- Leaves: All possible sorted orders (permutations) of the array of size \( n \)

**Question 2:** The maximum number of comparisons we might have to make (worst case) is the height of the tree. What is the height of a decision tree that is used to sort an array of size \( n \)?
We can construct a *branching diagram* or *decision tree* that shows the possible comparisons we might have to do.

**Binary Tree**

**Internal vertices:**
Comparisons

**Leaves:**
All possible sorted orders (permutations) of the array of size $n$

**Question 2:** The maximum number of comparisons we might have to make (worst case) is the height of the tree. What is the height of a decision tree that is used to sort an array of size $n$?

$log(n!)$
Height of the Decision Tree

If a binary tree has height $= k$, then it has $\leq 2^k$ leaves.
If a binary tree has height $< k$, then it has $< 2^k$ leaves.
If a binary tree has height $< \log(k)$, then it has $< k$ leaves.
Height of the Decision Tree

If a binary tree has height $= k$, then it has $\leq 2^k$ leaves.
If a binary tree has height $< k$, then it has $< 2^k$ leaves.
If a binary tree has height $< \log(k)$, then it has $< k$ leaves.

If a binary tree has $\geq k$ leaves, then it has height $\geq \log(k)$. ← Contrapositive
If a binary tree has $\geq n!$ leaves, then it has height $\geq \log(n!)$. 
Height of the Decision Tree

If a binary tree has height $= k$, then it has $\leq 2^k$ leaves.
If a binary tree has height $< k$, then it has $< 2^k$ leaves.
If a binary tree has height $< \log(k)$, then it has $< k$ leaves.

If a binary tree has $\geq k$ leaves, then it has has height $\geq \log(k)$. $\leftarrow$ Contrapositive
If a binary tree has $\geq n!$ leaves, then it has has height $\geq \log(n!)$.

This says the branching diagram for any sorting algorithm has height $\geq \log(n!)$.

Since the number of comparisons was the height of the tree, the worst case for any sorting algorithm is $\geq \log(n!)$ comparisons.
How big is log(n!)?

Lemma 1: For \( n > 1 \), \( \left( \frac{n}{2} \right)^{\frac{n}{2}} < n! < n^n \).

Proof:

\[
\begin{align*}
\log(n!) &= \log(n \cdot (n - 1) \cdot (n - 2) \cdots \frac{n}{2} \cdots 3 \cdot 2 \cdot 1) \\
&> \log \left( \frac{n}{2} \cdot \frac{n}{2} \cdot \frac{n}{2} \cdots \frac{n}{2} \right) \\
&= \left( \frac{n}{2} \right)^{\frac{n}{2}} \\
\log(n!) &= \log(n \cdot (n - 1) \cdot (n - 2) \cdots 3 \cdot 2 \cdot 1) \\
&< \log(n \cdot n \cdot n \cdots n \cdot n \cdot n) \\
&= (n^n)
\end{align*}
\]
How big is \( \log(n!) \)?

**Lemma 2**: \( \log(n!) \) is in \( \Theta(n \log n) \).

**Proof**:

\[
\left( \frac{n}{2} \right)^{\frac{n}{2}} < n! < n^n
\]

\[
\log \left( \frac{n}{2} \right)^{\frac{n}{2}} < \log(n!) < \log(n^n)
\]

\[
\frac{n}{2} \log \left( \frac{n}{2} \right) < \log(n!) < n \log(n)
\]
**Result:**

The best comparison-based sorting algorithms are $\Theta(n \log(n))$ like MergeSort.

It is impossible to have a comparison-based algorithm that does better than this in the worst case.
<table>
<thead>
<tr>
<th></th>
<th>Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Understanding and analyzing algorithms</td>
</tr>
<tr>
<td>2</td>
<td>Using graphs and graph algorithms</td>
</tr>
<tr>
<td>3</td>
<td>Using combinatorial reasoning to quantitatively analyze algorithms and systems</td>
</tr>
<tr>
<td>3.1</td>
<td>Basics of Counting: Product rule, Sum rule</td>
</tr>
<tr>
<td>3.2</td>
<td>Permutations, Combinations</td>
</tr>
<tr>
<td>3.3</td>
<td>Lower bound for comparison-based sorting</td>
</tr>
<tr>
<td>3.4</td>
<td>Fixed density binary strings</td>
</tr>
<tr>
<td>4</td>
<td>Using probability to algorithm analysis</td>
</tr>
</tbody>
</table>
Fixed Density Binary Strings
Example: Fixed Density Binary Strings

- We want to encode a length n binary string that we know has k ones.
- What kind of redundancy is there in the data?
- How can we use that to encode the data?

Thanks to Janine Tiefenbruck.