



# TOÁN RỜI RẠC

## Discrete Mathematics



Fall 2009

---

# Nguyễn Sơn Nguyễn

Bé m«n Khoa hăc M,y tÝnh

§'i hăc B,ch khoa Hư néi

***Tel: 0438696121 (Off), 0903210111 (Mob)***

 ***nghiand@it-hut.edu.vn***

## Đề nghị với các lớp trưởng

---

- Hãy gửi cho tôi danh sách lớp theo địa chỉ email đã nêu

# Toán rời rạc là gì?

---

- **What is discrete mathematics?**
  - Là bộ phận của toán học nghiên cứu các đối tượng rời rạc.
    - Rời rạc bao hàm ý các phần tử phân biệt hay không liên tục.
    - Các phép toán:
      - Tổ hợp: Đếm các đối tượng rời rạc
      - Các phép toán logic, quan hệ: nói lên mối quan hệ giữa các đối tượng rời rạc
    - Làm việc với: Các đối tượng rời rạc: tập hợp, mệnh đề.

# Định nghĩa hình thức - Wikipedia

---

- **Discrete mathematics**, sometimes called **finite mathematics**, is the study of mathematical structures that are fundamentally discrete, in the sense of not supporting or requiring the notion of continuity. Most, if not all, of the objects studied in finite mathematics are countable sets, such as the integers.
- Discrete mathematics has become popular in recent decades because of its applications to computer science. Concepts and notations from discrete mathematics are useful to study or express objects or problems in computer algorithms and programming languages. In some mathematics curricula, finite mathematics courses cover discrete mathematical concepts for business, while discrete mathematics courses emphasize concepts for computer science majors.
- Discrete mathematics usually includes :
  - logic - a study of reasoning
  - set theory - a study of collections of elements
  - number theory
  - combinatorics - a study of counting
  - graph theory
  - algorithmics - a study of methods of calculation
  - information theory
  - the theory of computability and complexity - a study on theoretical limitations on algorithms ...

# Nhập môn Toán rời rạc

---

- Các ứng dụng của TRR:
  - Formal Languages (computer languages)
  - Machine translation
  - Compiler Design
  - Artificial Intelligence
  - Relational Database Theory
  - Network Routing
  - Algorithm Design
  - many more (almost all areas of computer science)...

**A building block of computer science !**

# Nhập môn Toán rời rạc

---

- Các vấn đề chính được đề cập trong giáo trình này:
  - Cơ sở: logic, tập hợp, ánh xạ.
  - Lý thuyết tổ hợp (Combinatorial Theory)
    - Bài toán đếm
    - Bài toán tồn tại
    - Bài toán liệt kê
    - Bài toán tối ưu
  - Lý thuyết đồ thị (Graph theory):
    - Đồ thị, Đường đi, Liên thông
    - Biểu diễn đồ thị
    - Duyệt đồ thị
    - Các bài toán tối ưu trên đồ thị



## Tài liệu tham khảo

---

1. **Rosen K.H.** *Discrete Mathematics and its Applications*. McGraw - Hill Book Company, 2003.
2. **Johnsonbaugh R.** *Discrete Mathematics*. Prentice Hall Inc., N. J., 1997.
3. **Grimaldi R.P.** *Discrete and Combinatorial Mathematics (an Applied Introduction)*, Addison-Wesley, 5th edition, 2004.
4. **R. Graham, O. Patashnik, and D.E. Knuth.** *Concrete Mathematics*, Second Edition. Addison-Wesley, 1994.





## Tài liệu tham khảo

---

5. **Phan Đình Diệu.** *Lý thuyết ô tômat hữu hạn và thuật toán.* NXB ĐHTH-CN, Hà nội, 1977.
6. **Nguyễn Hữu Anh.** *Toán rời rạc,* NXB Giáo dục, 1999.
7. **Nguyễn Xuân Quỳnh.** *Cơ sở Toán rời rạc và ứng dụng.* NXB KHKT, Hà nội, 1996.
8. **Đỗ Đức Giáo.** *Toán rời rạc.* NXB KHKT, Hà nội, 2001.
9. **Hoàng Chúng.** *Đại cương về toán hữu hạn.* NXB Giáo dục, 1997.



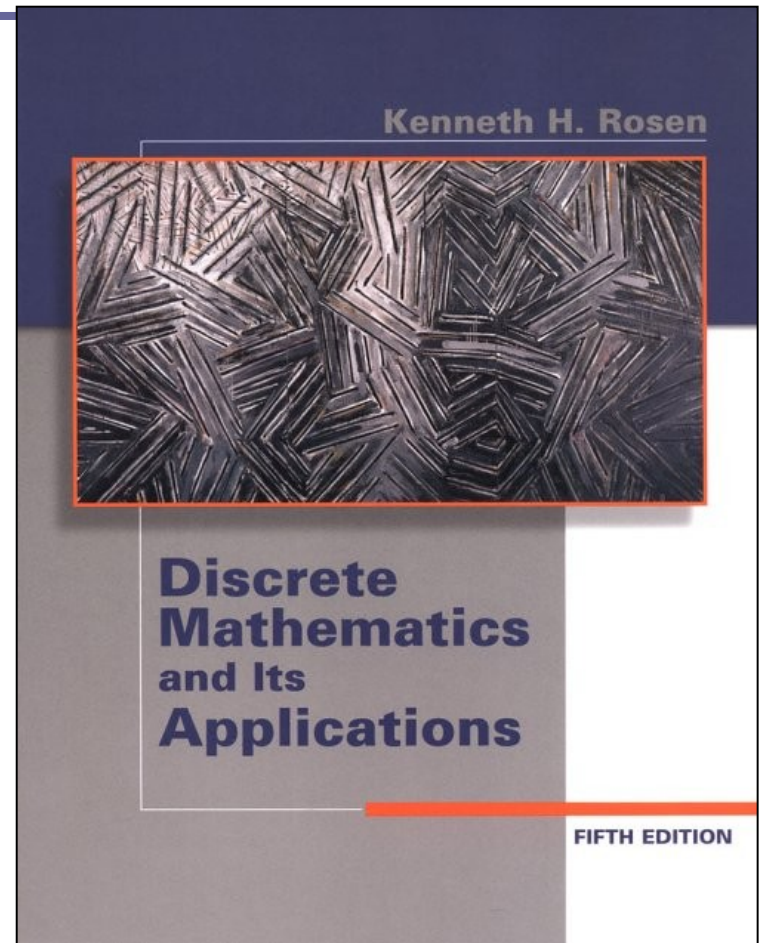
# Rosen's Book

<http://www.mhhe.com/math/advmath/rosen/index.mhtml>

**Rosen K.H.**

*Discrete Mathematics  
and its Applications. 5<sup>th</sup>  
Edition,*

McGraw - Hill Book  
Company, 2003.





# Table of Contents

---

Preface

To the Student

The Companion Web Site

## 1 The Foundations: Logic, Sets, and Functions

*Logic, Propositional Equivalences, Predicates and Quantifiers, Sets, Set Operations, Functions, Sequences and Summations, The Growth Functions*

## 2 The Fundamentals: Algorithms, the Integers, and Matrices

*Algorithms, Complexity of Algorithms, The Integers and Division, Integers and Algorithms, Applications of Number Theory, Matrices*

## 3 Mathematical Reasoning

*Methods of Proof, Mathematical Induction, Recursive Definitions, Recursive Algorithms, Program Correctness*

## 4 Counting

*The Basics of Counting, The Pigeonhole Principle, Permutations and Combinations, Discrete Probability, Probability Theory, Generalized Permutations and Combinations, Generating Permutations and Combinations*

## 5 Advanced Counting Techniques

*Recurrence Relations, Solving Recurrence Relations, Divide-and-Conquer Relations, Generating Functions, Inclusion-Exclusion, Applications of Inclusion-Exclusion*

## 6 Relations

*Relations and Their Properties, n-ary Relations and Their Applications, Representing Relations, Closures of Relations, Equivalence Relations, Partial Orderings*

## 7 Graphs

*Introduction to Graphs, Graph Terminology, Representing Graphs and Graph Isomorphism, Connectivity, Euler and Hamilton Paths, Shortest Path Problems, Planar Graphs, Graph Coloring*

## 8 Trees

*Introduction to Trees, Applications of Trees, Tree Traversal, Trees and Sorting, Spanning Trees, Minimum Spanning Trees*

## 9 Boolean Algebra

*Boolean Functions, Representing Boolean Functions, Logic Gates, Minimization of Circuits*

## 10 Modeling Computation

*Languages and Grammars, Finite-State Machines with Output, Finite-State Machines with No Output, Language Recognition, Turing Machines*

## Appendixes

**Suggested Readings**

**Solutions to Odd-Numbered Exercises**

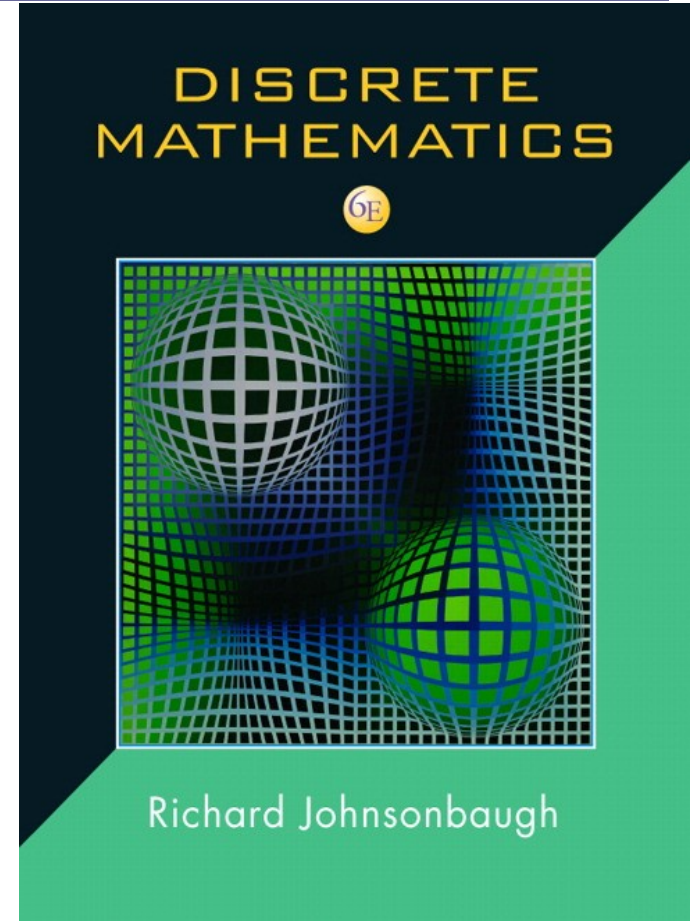
**Index of Biographies**

**Index**

# Johnsonbaugh Book



**Johnsonbaugh R.**  
*Discrete Mathematics.*  
Prentice Hall Inc.,  
N. J., 1997.



# Table of Contents



## 1 Logic and Proofs

- 1.1 Propositions
- 1.2 Conditional Propositions and Logical Equivalence
- 1.3 Quantifiers
- 1.4 Nested Quantifiers
- 1.5 Proofs
- 1.6 Resolution Proofs
- 1.7 Mathematical Induction
- 1.8 Strong Form of Induction and the Well-Ordering Property

## 2 The Language of Mathematics

- 2.1 Sets
- 2.2 Functions
- 2.3 Sequences and Strings

## 3 Relations

- 3.1 Relations
- 3.2 Equivalence Relations
- 3.3 Matrices of Relations
- 3.4 Relational Databases

## 4 Algorithms

- 4.1 Introduction
- 4.2 Examples of Algorithms
- 4.3 Analysis of Algorithms
- 4.4 Recursive Algorithms

## 5 Introduction to Number Theory

- 5.1 Divisors
- 5.2 Representations of Integers and Integer Algorithms
- 5.3 The Euclidean Algorithm
- 5.4 The RSA Public-Key Cryptosystem

## 6 Counting Methods and the Pigeonhole Principle

- 6.1 Basic Principles
- 6.2 Permutations and Combinations
- 6.3 Algorithms for Generating Permutations and Combinations
- 6.6 Generalized Permutations and Combinations
- 6.7 Binomial Coefficients and Combinatorial Identities
- 6.8 The Pigeonhole Principle

## 7 Recurrence Relations

- 7.1 Introduction
- 7.2 Solving Recurrence Relations
- 7.3 Applications to the Analysis of Algorithms

## 8 Graph Theory

- 8.1 Introduction
- 8.2 Paths and Cycles
- 8.3 Hamiltonian Cycles and the TSP
- 8.4 Shortest-Path Algorithm
- 8.5 Representations of Graphs

## 9 Trees

- 9.1 Introduction
- 9.2 Terminology and Characterizations of Trees
- 9.3 Spanning Trees
- 9.4 Minimal Spanning Trees
- 9.5 Binary Trees
- 9.6 Tree Traversals
- 9.7 Decision Trees and the Minimum Time for Sorting
- 9.8 Isomorphisms of Trees
- 9.9 Game Trees

## 10 Network Models

- 10.1 Introduction
- 10.2 A Maximal Flow Algorithm
- 10.3 The Max Flow, Min Cut Theorem
- 10.4 Matching

## 11 Boolean Algebras and Combinatorial Circuits

## 12 Automata, Grammars, and Languages

## 13 Computational Geometry

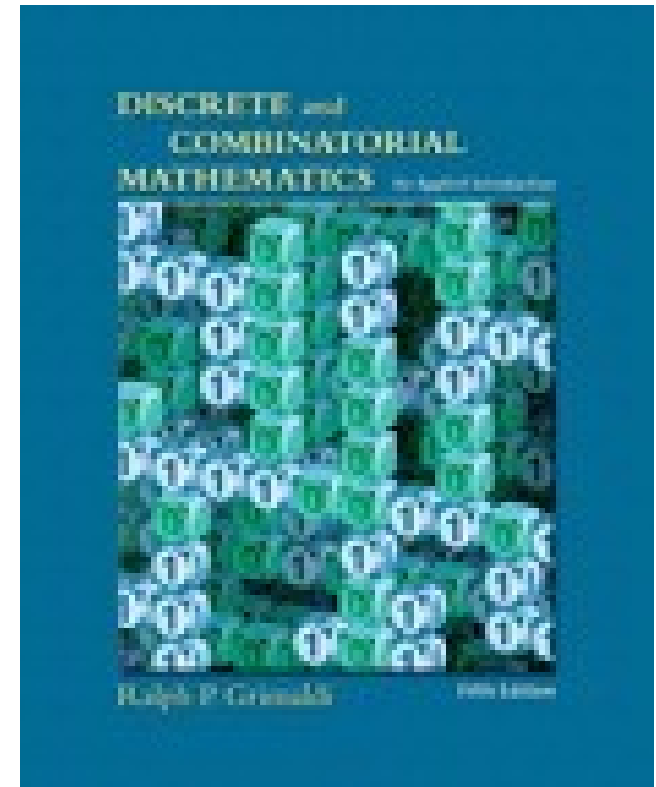
## Appendix

## Index

# Grimaldi's Book



**Grimaldi R.P.**  
*Discrete and  
Combinatorial  
Mathematics (an  
Applied Introduction),*  
Addison-Wesley, 5th  
edition, 2001.





# Table of Contents

## **PART 1. FUNDAMENTALS OF DISCRETE MATHEMATICS.**

### **1. Fundamental Principles of Counting.**

The Rules of Sum and Product.

Permutations. Combinations: The Binomial Theorem.

Combinations with Repetition.

### **2. Fundamentals of Logic.**

### **3. Set Theory.**

Sets and Subsets.

Set Operations and the Laws of Set Theory.

Counting and Venn Diagrams.

A First Word on Probability.

### **4. Properties of the Integers: Mathematical Induction.**

The Well-Ordering Principle: Mathematical Induction.

Recursive Definitions.

The Division Algorithm: Prime Numbers.

The Greatest Common Divisor: The Euclidean Algorithm.

The Fundamental Theorem of Arithmetic.

### **5. Relations and Functions.**

Cartesian Products and Relations.

Functions: Plain and One-to-One.

Onto Functions: Stirling Numbers of the Second Kind.

Special Functions.

The Pigeonhole Principle.

Function Composition and Inverse Functions.

Computational Complexity.

Analysis of Algorithms.

### **6. Languages: Finite State Machines.**

### **7. Relations: The Second Time Around.**

## **PART 2. FURTHER TOPICS IN ENUMERATION.**

### **8. The Principle of Inclusion and Exclusion.**

The Principle of Inclusion and Exclusion.

Derangements: Nothing Is in Its Right Place.

Rook Polynomials.

Arrangements with Forbidden Positions.

### **9. Generating Functions.**

Introductory Examples.

Definition and Examples: Calculational Techniques.

Partitions of Integers.

### **10. Recurrence Relations.**

## **PART 3. GRAPH THEORY AND APPLICATIONS.**

### **11. An Introduction to Graph Theory.**

Definitions and Examples.

Subgraphs, Complements, and Graph Isomorphism.

Vertex Degree: Euler Trails and Circuits.

Planar Graphs. Hamilton Paths and Cycles.

### **12. Trees.**

Definitions, Properties, and Examples.

Rooted Trees. Trees and Sorting.

Weighted Trees and Prefix Codes.

Biconnected Components and Articulation Points.

### **13. Optimization and Matching.**

Dijkstra's Shortest Path Algorithm.

Minimal Spanning Trees: The Algorithms of Kruskal and Prim.

Transport Networks: The Max-Flow Min-Cut Theorem.

Matching Theory.

## **PART 4. MODERN APPLIED ALGEBRA.**

### **14. Rings and Modular Arithmetic.**

### **15. Boolean Algebra and Switching Functions.**

### **16. Groups, Coding Theory, and Polya's Theory of Enumeration.**

### **17. Finite Fields and Combinatorial Designs.**



# Graham, Knuth, Patashnik's Book

---

**Ronald L. Graham**

**Donald E. Knuth**

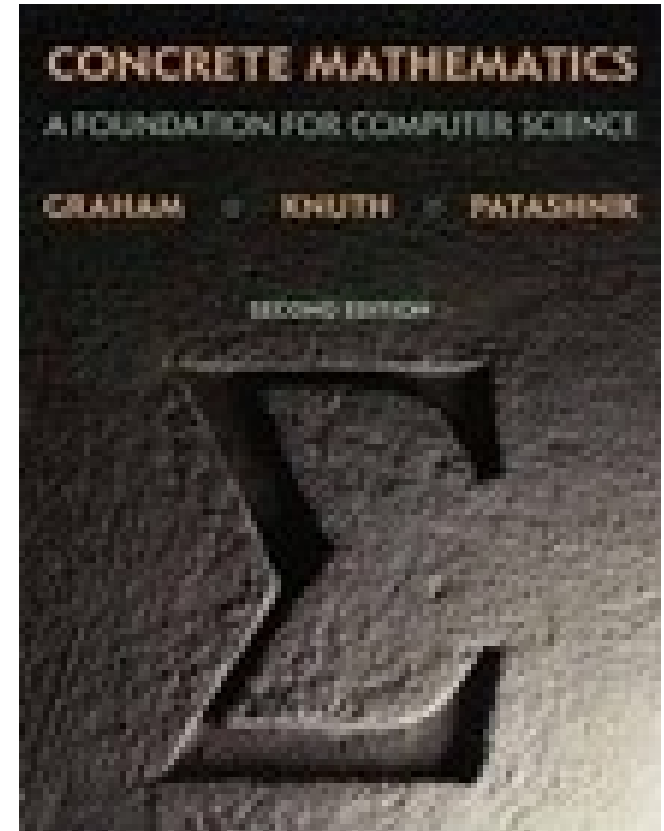
**Oren Patashnik**

*Concrete Mathematics:*

*A Foundation for Computer  
Science,*

Addison-Wesley Professional

1994, 672 pp







# Table of Contents

## 1. Recurrent Problems.

The Tower of Hanoi.  
Lines in the Plane.  
The Josephus Problem.

## 2. Sums.

Notation.  
Sums and Recurrences.  
Manipulation of Sums.  
Multiple Sums.  
General Methods.  
Finite and Infinite Calculus.  
Infinite Sums.

## 3. Integer Functions.

Floors and Ceilings.  
Floor/Ceiling Applications.  
Floor/Ceiling Recurrences.  
'mod': The Binary Operation.  
Floor/Ceiling Sums.

## 4. Number Theory.

Divisibility.  
Factorial Factors.  
Relative Primality.  
'mod': The Congruence Relation.  
Independent Residues.  
Additional Applications.  
Phi and Mu.

## 5. Binomial Coefficients.

Basic Identities. Basic Practice. Tricks of the Trade.  
Generating Functions.  
Hypergeometric Functions.  
Hypergeometric Transformations.

## 6. Special Numbers.

Stirling Numbers.  
Eulerian Numbers.  
Harmonic Numbers.  
Harmonic Summation.  
Bernoulli Numbers.  
Fibonacci Numbers.  
Continuants.

## 7. Generating Functions.

Domino Theory and Change.  
Basic Maneuvers.  
Solving Recurrences.  
Special Generating Functions.  
Convolutions.  
Exponential Generating Functions.  
Dirichlet Generating Functions.

## 8. Discrete Probability.

Definitions.  
Mean and Variance.  
Probability Generating Functions.  
Flipping Coins.  
Hashing.

## 9. Asymptotics.

A Hierarchy.  
O Notation.  
O Manipulation.  
Two Asymptotic Tricks.  
Euler's Summation Formula.  
Final Summations.

## A. Answers to Exercises.

## B. Bibliography.

## C. Credits for Exercises.

## Tài liệu tham khảo chính

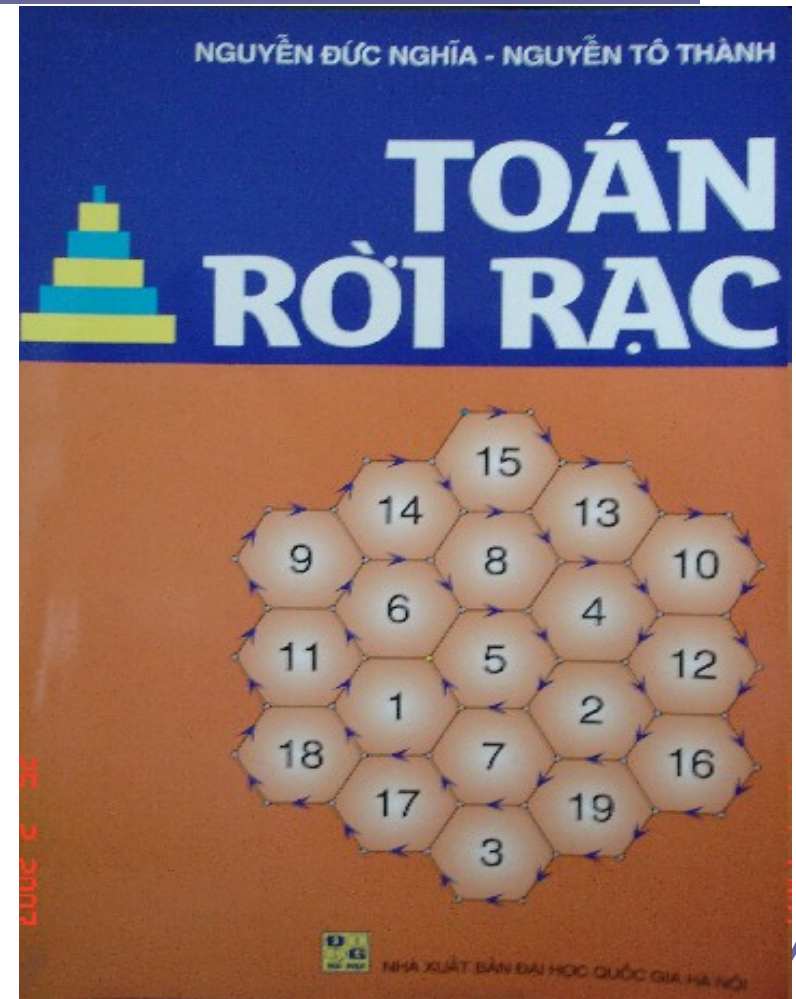


**Nguyễn Đức Nghĩa,  
Nguyễn Tô Thành**

**TOÁN RỜI RẠC**

**(in lần thứ ba)**

Nhà xuất bản Đại học  
Quốc gia Hà nội, 2003,  
290 trang





# Mục lục

## Phần I. Lý thuyết Tập hợp

### Chương 1. Mở đầu

- 1.1 Sơ lược về tổ hợp
- 1.2 Nhắc lại lý thuyết tập hợp
- 1.3 Một số nguyên lý cơ bản
- 1.4 Các cấu hình tổ hợp đơn giản

### Chương 2. Bài toán đếm

- 2.1 Giới thiệu bài toán
- 2.2 Nguyên lý bù trừ
- 2.3 Quy về các bài toán đơn giản
- 2.4 Công thức truy hồi
- 2.5 Liệt kê

### Chương 3. Bài toán tồn tại

- 3.1 Giới thiệu bài toán
- 3.2 Phương pháp phản chứng
- 3.3 Nguyên lý Dirichlet
- 3.4 Hệ đại diện phân biệt

### Chương 4. Bài toán liệt kê

- 4.1 Giới thiệu bài toán
- 4.2 Thuật toán và độ phức tạp tính toán
- 4.3 Phương pháp sinh
- 4.4 Thuật toán quay lui

### Chương 5. Bài toán tối ưu

- 5.1 Phát biểu bài toán
- 5.2 Các thuật toán duyệt
- 5.3 Thuật toán nhánh cận giải bài toán người du lịch
- 5.4 Bài toán lập lịch gia công trên hai máy

## Phần 2. Lý thuyết Đồ thị

### Chương 1. Các khái niệm cơ bản của lý thuyết đồ thị

- 1.1 Định nghĩa đồ thị
- 1.2 Các thuật ngữ cơ bản
- 1.3 Đường đi, Chu trình, Đồ thị liên thông
- 1.4 Một số dạng đồ thị đặc biệt

### Chương 2. Biểu diễn đồ thị trên máy tính

- 2.1 Ma trận kề, Ma trận trọng số, 2.2 Danh sách cạnh 2.3 Danh sách kề

### Chương 3. Các thuật toán tìm kiếm trên đồ thị và ứng dụng

- 3.1 Tìm kiếm theo chiều sâu trên đồ thị
- 3.2 Tìm kiếm theo chiều rộng trên đồ thị
- 3.3 Tìm đường đi và kiểm tra tính liên thông

### Chương 4. Đồ thị Euler và đồ thị Hamilton

- 4.1 Đồ thị Euler 4.2 Đồ thị Hamilton

### Chương 5. Cây và cây khung của đồ thị

- 5.1 Cây và các tính chất của cây
- 5.2 Cây khung của đồ thị
- 5.3 Xây dựng tập các chu trình cơ bản của đồ thị
- 5.4 Bài toán cây khung nhỏ nhất

### Chương 6. Bài toán đường đi ngắn nhất

### Chương 7. Bài toán luồng cực đại trong mạng

## Phần 3. Hàm Logic

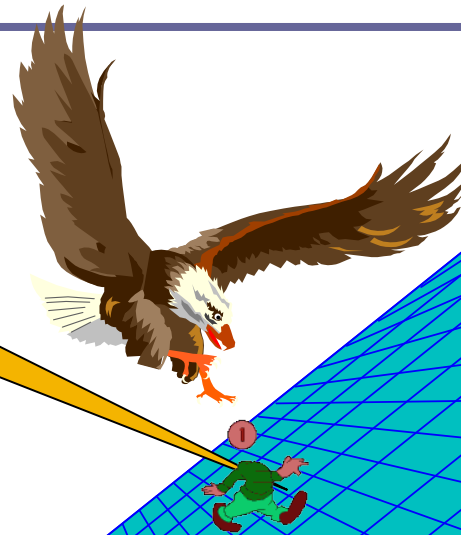
### Chương 1. Mở đầu

### Chương 2. Dạng tuyến chuẩn tắc của hàm đại số logic

### Chương 3. Thuật toán tìm dạng tuyến chuẩn tắc tối thiểu

### Tài liệu tham khảo

Questions?







# Ứng dụng: Ngôn ngữ hình thức

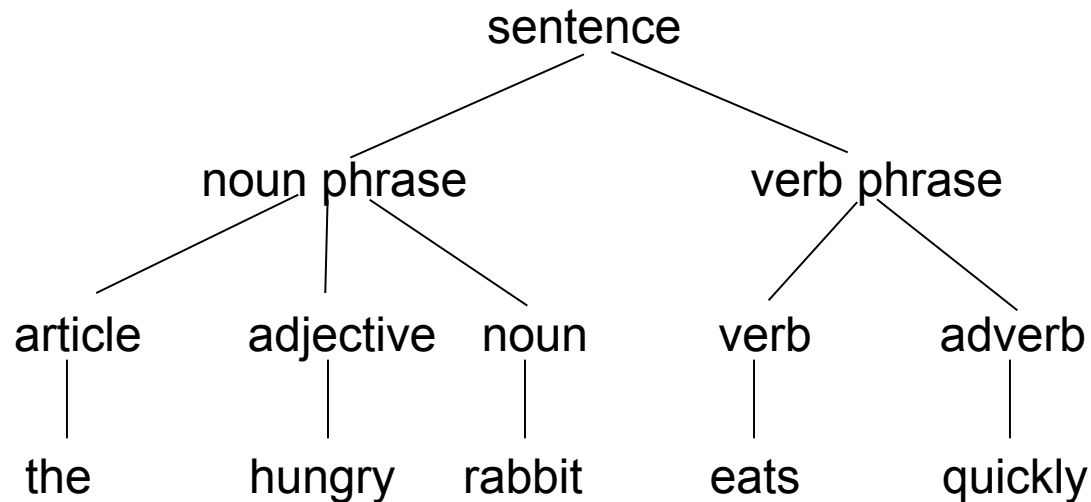
## Formal Language

---

- Formal language:
  - Ngôn ngữ được sinh bởi ngữ pháp (grammars)
  - Grammars:
    - Sinh ra các từ (words) của ngôn ngữ
    - Xác định một từ có thuộc ngôn ngữ hay không
  - Từ (Words):
    - Có thể tổ hợp bằng nhiều cách
    - Ngữ pháp cho biết tổ hợp từ có là một câu hợp lệ (valid sentence) hay không
  - Ứng dụng:
    - Thiết kế các Ngôn ngữ lập trình và Chương trình dịch (Programming Languages and Compilers)

# Ứng dụng: Ngôn ngữ hình thức

- Ví dụ 1:
  - Ngôn ngữ tự nhiên: English
    - Có phải “the hungry rabbits eats quickly” là câu tiếng Anh?
    - Cây cú pháp của câu (Derivation tree of the sentence):





# Ứng dụng: Ngôn ngữ hình thức

- Ví dụ 2:
  - Bài toán điển hình trong xây dựng Chương trình dịch.
  - Xác định từ **cbab** có thuộc ngôn ngữ sinh bởi ngữ pháp  $G = (V, T, S, P)$ , trong đó:
    - $V = \{ a, b, c, A, B, C, S \}$
    - $T = \{ a, b, c \}$
    - $S$  là ký tự đầu tiên (starting symbol)
    - $P$  là các luật sản xuất (productions):

$S \rightarrow AB$	$A \rightarrow Ca$	$B \rightarrow Ba$
$B \rightarrow Cb$	$B \rightarrow b$	$C \rightarrow cb$
$C \rightarrow b$		

## Ứng dụng: Ngôn ngữ hình thức

- Ví dụ 2 (tiếp tục):

- Giải:

- Bắt đầu bởi  $S$  và tìm cách dẫn ra  $cbab$  sử dụng dãy các luật sản xuất.

- Do chỉ có một luật bắt đầu với  $S$ , ta phải bắt đầu với

$$S \Rightarrow AB$$

- Sử dụng luật đối với  $A$  để thu được:

$$S \Rightarrow AB \Rightarrow CaB$$

- Sử dụng luật đối với  $C \rightarrow cb$  vì  $cbab$  bắt đầu bởi  $cb$ :

$$S \Rightarrow AB \Rightarrow CaB \Rightarrow cbaB$$

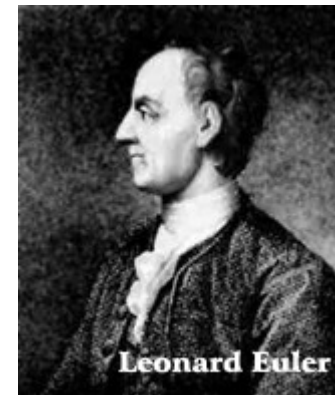
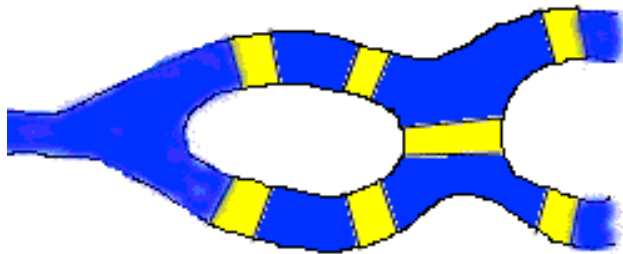
- Cuối cùng sử dụng luật đối với  $B \rightarrow b$ :

$$S \Rightarrow AB \Rightarrow CaB \Rightarrow cbaB \Rightarrow cbab$$

# Ứng dụng: Graph Theory

- **Ví dụ 3**

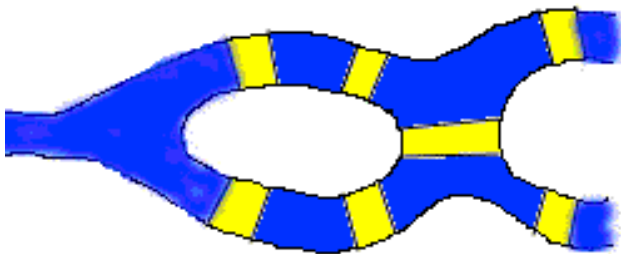
- Bài toán về 7 cái cầu (Konigsberg 7-bridge problem)
  - Königsberg là thành phố của Nga
  - Có 4 vùng đất, và 7 cái cầu nối chúng
  - Euler giải được bài toán năm 1736; là khởi nguồn của lý thuyết đồ thị



## Ứng dụng: Graph Theory

---

- Bài toán: Vẽ đường đi (hoặc vòng kín) bởi bút chì sao cho có thể đi qua mỗi cái cầu đúng một lần mà không được nhấc đầu bút khỏi mặt giấy (vẽ một nét)



## Ứng dụng: Lý thuyết tập hợp (Set Theory)

---

- Có phải số lượng số nguyên là nhiều hơn số lượng số nguyên dương?
- Có phải số lượng số nguyên là nhiều hơn số lượng số thực?

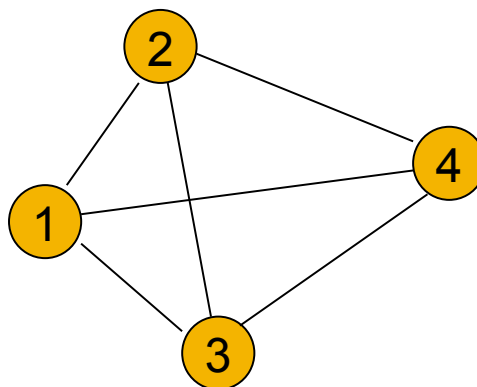
# Ứng dụng: Lý thuyết độ phức tạp (Complexity Theory)

---

- Ví dụ 5:
  - Bài toán người du lịch (The Traveling Salesman Problem)
  - Có ứng dụng quan trọng trong
    - Thiết kế mạch (circuit design)
    - Hướng lộ trên mạng (network routing)
    - và nhiều bài toán trong tin học khác
  - Cho:
    - $n$  thành phố  $c_1, c_2, \dots, c_n$
    - khoảng cách giữa thành phố  $i$  và  $j$  là  $d_{ij}$
  - Tìm hành trình ngắn nhất.

## Ứng dụng: Lý thuyết độ phức tạp

- Ví dụ 5 (tiếp):



- Có bao nhiêu hành trình khác nhau?
  - Thành phố đầu tiên có thể chọn bởi  $n$  cách,
    - thành phố thứ hai,  $n-1$  cách,
    - thành phố thứ ba,  $n-2$  cách,
    - v.v...
  - # hành trình =  $n (n-1) (n-2) \dots (2) (1) = n!$  (Tổ hợp)
- Tính độ dài của một hành trình đòi hỏi  $n-1$  phép cộng.
  - Tổng số phép cộng =  $(n-1) \times n!$  (Quy tắc nhân - Rule of Product)

## Ứng dụng: Lý thuyết độ phức tạp

- Ví dụ 5 (tiếp):
  - Giả sử có PC tốc độ rất cao:
    - 1 GHz = 1,000,000,000 ops/sec  
 $\Rightarrow$  1 flop = 1 nanosecond  
 $= 10^{-9}$  sec.
    - Nếu  $n=8$ ,  $T(n) = 7 \cdot 8! = 282,240$  flops  $\ll$  1 second.
    - TUY NHIÊN . . . . .
    - Nếu  $n=50$ ,  $T(n) = 49 \cdot 50! = 1.48 \cdot 10^{66}$   
 $= 1.49 \cdot 10^{57}$  seconds  
 $= 4.73 \cdot 10^{49}$  năm.
    - ... quá lâu. Không ai trong số chúng ta có thể chờ được thời điểm kết thúc.
    - Có rất nhiều bài toán mà chúng ta còn chưa biết liệu có thuật toán hiệu quả để giải hay không!



## Ứng dụng: Lý thuyết độ phức tạp

---

- Ví dụ 5 (tiếp):
  - Chúng ta có thể làm gì ?
    - Không có thời gian để chờ Do not waste time unless you are a genius to save the world
    - Mục đích khiêm tốn hơn
      - Với xác suất 90%, có thể tìm được hành trình tối ưu
      - Thuật toán tìm hành trình không tồi hơn 1.1 lần hành trình tối ưu