

Olympic Tin Học Sinh Viên OLP & ACM-ICPC

Nguyễn Quân Bá Hồng*

Ngày 8 tháng 3 năm 2025

Tóm tắt nội dung

This text is a part of the series *Some Topics in Advanced STEM & Beyond*:

URL: https://nqbh.github.io/advanced_STEM/.

Latest version:

- *Olympic Tin Học Sinh Viên OLP & ICPC*.
PDF: URL: https://github.com/NQBH/advanced_STEM_beyond/blob/main/OLP_ICPC/NQBH_OLP_ICPC.pdf.
T_EX: URL: https://github.com/NQBH/advanced_STEM_beyond/blob/main/OLP_ICPC/NQBH_OLP_ICPC.tex.
- Codes:
 - C: https://github.com/NQBH/advanced_STEM_beyond/tree/main/OLP_ICPC/C.
 - C++: https://github.com/NQBH/advanced_STEM_beyond/tree/main/OLP_ICPC/C++.
 - Python: https://github.com/NQBH/advanced_STEM_beyond/tree/main/OLP_ICPC/Python.

Mục lục

1 Basic Competitive Programming – Lập Trình Thi Đấu Cơ Bản	2
1.1 Various types of inputs & outputs – Các dạng dữ liệu đầu vào & đầu ra	2
1.2 Repeat/Loop – Lặp	2
1.3 String data – Kiểu dữ liệu chuỗi	2
1.4 Array data – Kiểu dữ liệu mảng	2
2 Olympic Tin THCS & THPT	3
3 VNOI	3
4 CSES Problem Set	3
4.1 Introductory Problems	3
4.2 Dynamic Programming	4
4.3 Graph Algorithms	4
4.4 Range Queries	4
4.5 Mathematics	4
4.6 String Algorithms	4
4.7 Geometry	4
4.8 Advanced Techniques	4
4.9 Additional Problems	4
5 OLP	4
6 ICPC	4
7 Miscellaneous	4
7.1 Contributors	4
7.2 Donate or Buy Me Coffee	4
7.3 See also	5
Tài liệu	5

*A Scientist & Creative Artist Wannabe. E-mail: nguyenquanbahong@gmail.com. Bến Tre City, Việt Nam.

1 Basic Competitive Programming – Lập Trình Thi Đấu Cơ Bản

Resources – Tài nguyên.

1. [Laa20]. ANTTI LAAKSONEN. *Guide to Competitive Programming: Learning & Improving Algorithms Through Contests*.
2. [Thu+21a]. TRẦN ĐAN THƯ, NGUYỄN THANH PHƯƠNG, ĐINH BÁ TIẾN, TRẦN MINH TRIẾT. *Nhập Môn Lập Trình*.
3. [Thu+21b]. TRẦN ĐAN THƯ, NGUYỄN THANH PHƯƠNG, ĐINH BÁ TIẾN, TRẦN MINH TRIẾT, ĐẶNG BÌNH PHƯƠNG. *Kỹ Thuật Lập Trình*.
4. [TTK21]. TRẦN ĐAN THƯ, ĐINH BÁ TIẾN, NGUYỄN TẤN TRẦN MINH KHANG. *Phương Pháp Lập Trình Hướng Đối Tượng*.

1.1 Various types of inputs & outputs – Các dạng dữ liệu đầu vào & đầu ra

To compile a C++ program in Linux, run in Terminal:

```
$ g++ -O2 -Wall program_name.cpp -o program_name
$ ./program_name
```

or if you want to transfer input file into it & print output into Terminal screen:

```
$ ./program_name < program_name.inp
```

or if you want to transfer input file into it & print output into a file:

```
$ ./program_name < program_name.inp > program_name.out
```

See, e.g., [Laa20].

- Geeks4Geeks/std::endl vs. \n in C++: <https://www.geeksforgeeks.org/endl-vs-n-in-cpp/>.
- i++ vs. ++i: [StackOverflow/Is there a performance difference between i++ & ++i in C?](https://stackoverflow.com/questions/1038483/whats-the-difference-between-i-i-and-i-i)

1.2 Repeat/Loop – Lặp

1.3 String data – Kiểu dữ liệu chuỗi

1.4 Array data – Kiểu dữ liệu mảng

Về mặt toán học, kiểu dữ liệu mảng là dãy số hữu hạn $(a_i)_{i=1}^n = (a_1, a_2, \dots, a_n)$. Về mặt Tin học, kiểu dữ liệu mảng được ký hiệu bởi $a[1..n]$.

1 ([Đúc22], 141., pp. 140–141: Count digit – Đếm chữ số). Cho dãy số n số nguyên dương $A[1..n]$ & 1 chữ số k . Đếm số lần xuất hiện chữ số k trong dãy A đã cho. E.g., với dãy $A[] = (11, 12, 13, 14, 15)$, thì chữ số $k = 1$ xuất hiện 6 lần trong dãy A .

Input. Dòng đầu tiên của đầu vào chứa số nguyên $T \in \mathbb{N}^*$ cho biết số bộ dữ liệu cần kiểm tra. Mỗi bộ dữ liệu gồm: (i) Dòng đầu chứa lần lượt $n, k \in \mathbb{N}$ là số phần tử trong dãy $A[]$ & chữ số k . (ii) Dòng thứ 2 chứa n số nguyên cách nhau 1 dấu cách, mô tả các phần tử của dãy A .

Output. Ứng với mỗi bộ dữ liệu, in ra 1 dòng chứa kết quả của bài toán tương ứng với bộ dữ liệu đầu vào đó.

Constraint. $1 \leq T \leq 100, 1 \leq n \leq 100, 0 \leq k \leq 9, 1 \leq A[i] \leq 1000, \forall i = 1, \dots, n$.

- Input: https://github.com/NQBH/advanced_STEM_beyond/blob/main/OLP_ICPC/input/count_digit.inp.
- Output: https://github.com/NQBH/advanced_STEM_beyond/blob/main/OLP_ICPC/output/count_digit.out.
- Python: https://github.com/NQBH/advanced_STEM_beyond/blob/main/OLP_ICPC/Python/count_digit.py.
- C++: ?

2 ([Đúc22], 141., pp. 140–141: Count digit – Đếm chữ số). Cho dãy số nguyên $a[1], a[2], \dots, a[n]$. Thực hiện nhiệm vụ: Chia dãy thành 2 phần trái & phải, trong đó phần trái gồm $\frac{n}{2}$ phần tử đầu tiên & phần phải gồm các phần tử còn lại. Tính tổng các phần tử của mỗi phần, cuối cùng tính & in ra tích 2 tổng tìm được.

Input. Dòng đầu tiên của đầu vào chứa $t \in \mathbb{N}^*$ cho biết số bộ dữ liệu cần kiểm tra. Mỗi bộ dữ liệu gồm: (i) Dòng đầu chứa $n \in \mathbb{N}^*$ cho biết số phần tử của dãy. (ii) Dòng 2 chứa n số nguyên cách nhau bởi dấu cách, là các phần tử của dãy.

Output. Ứng với mỗi bộ dữ liệu, in ra 1 dòng chứa kết quả của bài toán tương ứng với bộ dữ liệu đầu vào đó.

Constraint. $1 \leq t \leq 100, 1 \leq n \leq 100, 1 \leq A[i] \leq 100, \forall i = 1, \dots, n$.

- Input: https://github.com/NQBH/advanced_STEM_beyond/blob/main/OLP_ICPC/input/prod_left_right_sums.inp.
- Output: https://github.com/NQBH/advanced_STEM_beyond/blob/main/OLP_ICPC/output/prod_left_right_sums.out.
- Python: https://github.com/NQBH/advanced_STEM_beyond/blob/main/OLP_ICPC/Python/prod_left_right_sums.py.
- C++: ?

2 Olympic Tin THCS & THPT

3 ([Tru23], HSG12 Tp. Hà Nội 2020–2021, Prob. 1, p. 80: Find mid – Tìm giữa). (a) Cho $l, r \in \mathbb{N}^*$. Tìm $m \in [l, r] \cap \mathbb{N}^*$ để chênh lệch giữa tổng các số nguyên liên tiếp từ l đến m & tổng các số nguyên liên tiếp từ $m + 1$ đến r là nhỏ nhất. (b) Mở rộng cho $l, r \in \mathbb{Z}$. (c*) Thay tổng bởi tổng bình phương, tổng lập phương, tổng lũy thừa bậc $a \in \mathbb{R}$.

Input. 2 số $l, r \in \mathbb{N}^*$, $l < r \leq 10^9$.

Output. Gồm 1 số nguyên duy nhất là m thỏa mãn.

Limits. Subtask 1: 60% các test có $l < r \leq 10^3$. Subtask 2: 40% các test còn lại có $l < r \leq 10^9$.

- Input: https://github.com/NQBH/advanced_STEM_beyond/blob/main/OLP_ICPC/input/find_mid.inp.
- Output:
- C++: https://github.com/NQBH/advanced_STEM_beyond/blob/main/OLP_ICPC/C++/find_mid.cpp.
- Python: https://github.com/NQBH/advanced_STEM_beyond/blob/main/OLP_ICPC/Python/find_mid.py.

3 VNOI

4 (gcd in Pascal triangle – ƯCLN trong tam giác Pascal, <https://oj.vnoi.info/problem/gpt>). Tam giác Pascal là 1 cách sắp xếp hình học của các hệ số nhị thức vào 1 tam giác. Hàng thứ $n \in \mathbb{N}$ của tam giác bao gồm các hệ số trong khai triển của đa thức $f(x, y) = (x + y)^n$. I.e., phần tử tại cột thứ k , hàng thứ n của tam giác Pascal là $C_n^k = \binom{n}{k}$, i.e., tổ hợp chập k của n phần tử $0 \leq k \leq n$. Cho $n \in \mathbb{N}$. Tính GPT(n) là ƯCLN của các số nằm giữa 2 số 1 trên hàng thứ n của tam giác Pascal.

Input. Dòng đầu ghi T là số lượng test. T dòng tiếp theo, mỗi dòng ghi 1 số nguyên n .

Output. Gồm T dòng, mỗi dòng ghi GPT(n) tương ứng.

Constraint. $1 \leq T \leq 20$, $2 \leq n \leq 10^9$.

Phân tích. Công thức khai triển nhị thức Newton: $(a + b)^n = \sum_{i=0}^n C_n^i a^{n-i} b^i$, $\forall n \in \mathbb{N}$, see, e.g., [Wikipedia/binomial theorem](https://en.wikipedia.org/wiki/Binomial_theorem). Cần tính $\gcd(\{C_n^i; 1 \leq i \leq n-1\}) = \gcd(C_n^1, C_n^2, \dots, C_n^{n-1})$. Chú ý mỗi hàng của tam giác Pascal có tính chất đối xứng nên chỉ cần xét “1 nửa” là đủ. Cụ thể hơn: $C_n^k = C_n^{n-k}$, $\forall k \in \mathbb{N}$, $k \leq n$, nên

$$\{C_n^1, \dots, C_n^{n-1}\} = \{C_n^1, \dots, C_n^{\lfloor \frac{n}{2} \rfloor}\} = \begin{cases} \{C_n^1, \dots, C_n^{\frac{n-1}{2}}\} & \text{if } n \not\equiv 2, \\ \{C_n^1, \dots, C_n^{\frac{n}{2}}\} & \text{if } n \equiv 2, \end{cases}$$

nên thay vì xét $i = 1, \dots, n-1$, chỉ cần xét $i = 1, \dots, \lfloor \frac{n}{2} \rfloor$ là đủ.

Theorem 1.

$$\gcd\{C_n^i\}_{i=1}^{n-1} = \begin{cases} p & \text{if } n = p^k \text{ for some prime } p \text{ & some } n \in \mathbb{N}^*, \\ 1 & \text{if } n \neq p^k \text{ for all prime } p \text{ & any } n \in \mathbb{N}^*. \end{cases}$$

See also, e.g.:

- [Mathematics StackExchange/gcd of binomial coefficients](https://math.stackexchange.com/questions/1111111/gcd-of-binomial-coefficients).

4 CSES Problem Set

Link: <https://cses.fi/problemset/>.

4.1 Introductory Problems

Problem 1 (CSES). There are n concert tickets available, each with a certain price. Then, m customers arrive, one after another. Each customer announces the maximum price they are willing to pay for a ticket, & after this, they will get a ticket with nearest possible price such that it does not exceed the maximum price.

Input. 1st input line contains $n, m \in \mathbb{N}$: number of tickets & number of customers. The next line contains n integers h_1, h_2, \dots, h_n : the price of each ticket. The last line contains m integers t_1, t_2, \dots, t_m : the maximum price for each customer in the order they arrive.

Output. Print, for each customer, the price that they will pay for their ticket. After this, ticket cannot be purchased again. If a customer cannot get any ticket, print -1 .

Constraints. $1 \leq m, n \leq 2 \cdot 10^5$, $1 \leq h_i, t_i \leq 10^9$.

4.2 Dynamic Programming

Resources – Tài nguyên.

1. [Ber05; Ber17]. DIMITRI P. BERTSEKAS. *Dynamic Programming & Optimal Control. Vol. I.* 3e. 4e (can't download yet).
2. [Ber07; Ber12] DIMITRI P. BERTSEKAS. *Dynamic Programming & Optimal Control. Vol. II.* 3e. 4e (can't download yet).

4.3 Graph Algorithms

4.4 Range Queries

4.5 Mathematics

Problem 2 (CSES/Josephus Queries, <https://cses.fi/problemset/task/2164>). Consider a game where there are $n \in \mathbb{N}^*$ children, numbered $1, 2, \dots, n$, in a circle. During the game, every 2nd child is removed from circle, until there are no children left. Task: process q queries of the form: “when there are n children, who is the k th child that will be removed?”

- Input. The 1st input line has an integer q : the number of queries. After this, there are q lines that describe the queries. Each line has 2 integers n, k : the number of children & the position of the child.
- Output. Print q integers: the answer for each query.

It seems to me that Jack97 (nickname: abortion_grandmaster) proposed this problem.

Codes:

- C++: https://github.com/NQBH/advanced_STEM_beyond/blob/main/OLP_ICPC/C%2B%2B/gcd_Pascal_triangle.cpp.

Problem 3 (CSES/Dice Probability, <https://cses.fi/problemset/task/1725>). Throw a dice $n \in \mathbb{N}^*$ times, & every throw produces an outcome between 1 & 6. What is the probability that the sum of outcomes is between $a, b \in \mathbb{Z}$?

- Input. The only input line contains 3 integers $n, a, b \in \mathbb{N}^*$.
- Output. Print probability rounded to 6 decimal places (rounding half to even).
- Constraints. $1 \leq n \leq 100, 1 \leq a \leq b \leq 6n$.
- Example. Input: 2 9 10. Output: 0.194444.

Phân tích. Gọi n outcomes là $a_1, \dots, a_n \in \{1, \dots, 6\}$. Sum of outcomes: $S := \sum_{i=1}^n a_i \in \{n, \dots, 6n\}$.

4.6 String Algorithms

4.7 Geometry

4.8 Advanced Techniques

4.9 Additional Problems

5 OLP

6 ICPC

7 Miscellaneous

7.1 Contributors

1. VÕ NGỌC TRÂM ANH. C++ codes.
2. DẶNG PHÚC AN KHANG. C++ codes.
 - *Combinatorics & Number Theory in Competitive Programming – Tổ Hợp & Lý Thuyết Số trong Lập Trình Thi Đấu.*
3. NGUYỄN LÊ ANH KHOA: C++ codes.
4. PHAN VINH TIẾN. C++ codes.

7.2 Donate or Buy Me Coffee

Donate (but do not donut) or buy me some coffee via NQBH's bank account information at https://github.com/NQBH/publication/blob/master/bank/NQBH_bank_account_information.

7.3 See also

1. *Vietnamese Mathematical Olympiad for High School- & College Students (VMC) – Olympic Toán Học Học Sinh & Sinh Viên Toàn Quốc.*

PDF: URL: https://github.com/NQBH/advanced_STEM_beyond/blob/main/VMC/NQBH_VMC.pdf.

TEX: URL: https://github.com/NQBH/advanced_STEM_beyond/blob/main/VMC/NQBH_VMC.tex.

- Codes:
 - C++ code: https://github.com/NQBH/advanced_STEM_beyond/tree/main/VMC/C++.
 - Python code: https://github.com/NQBH/advanced_STEM_beyond/tree/main/VMC/Python.
- Resource: https://github.com/NQBH/advanced_STEM_beyond/tree/main/VMC/resource.
- Figures: https://github.com/NQBH/advanced_STEM_beyond/tree/main/VMC/figure.

Tài liệu

- [Ber05] Dimitri P. Bertsekas. *Dynamic programming and optimal control. Vol. I.* Third. Athena Scientific, Belmont, MA, 2005, pp. xvi+543. ISBN: 1-886529-26-4.
- [Ber07] Dimitri P. Bertsekas. *Dynamic programming and optimal control. Vol. II.* Third. Athena Scientific, Belmont, MA, 2007, p. 445.
- [Ber12] Dimitri P. Bertsekas. *Dynamic programming and optimal control. Vol. II. Approximate dynamic programming.* Fourth. Athena Scientific, Belmont, MA, 2012, pp. xvii+694. ISBN: 978-1-886529-44-1; 1-886529-44-2; 1-886529-08-6.
- [Ber17] Dimitri P. Bertsekas. *Dynamic programming and optimal control. Vol. I.* Fourth. Athena Scientific, Belmont, MA, 2017, pp. xix+555. ISBN: 978-1-886529-43-4; 1-886529-43-4; 1-886529-08-6.
- [Đức22] Nguyễn Tiến Đức. *Tuyển Tập 200 Bài Tập Lập Trình Bằng Ngôn Ngữ Python.* Nhà Xuất Bản Đại Học Thái Nguyên, 2022, p. 327.
- [Laa20] Antti Laaksonen. *Guide to Competitive Programming: Learning & Improving Algorithms Through Contests.* 2nd edition. Undergraduate Topics in Computer Science. Springer, 2020, pp. xv+309.
- [Thư+21a] Trần Đan Thư, Nguyễn Thanh Phương, Đinh Bá Tiến, and Trần Minh Triết. *Nhập Môn Lập Trình.* Nhà Xuất Bản Khoa Học & Kỹ Thuật, 2021, p. 427.
- [Thư+21b] Trần Đan Thư, Nguyễn Thanh Phương, Đinh Bá Tiến, Trần Minh Triết, and Đặng Bình Phương. *Kỹ Thuật Lập Trình.* Nhà Xuất Bản Khoa Học & Kỹ Thuật, 2021, p. 526.
- [Tru23] Vương Thành Trung. *Tuyển Tập Đề Thi Học Sinh Giỏi Cấp Tỉnh Trung Học Phổ Thông Tin Học.* Tài liệu lưu hành nội bộ, 2023, p. 235.
- [TTK21] Trần Đan Thư, Đinh Bá Tiến, and Nguyễn Tấn Trần Minh Khang. *Phương Pháp Lập Trình Hướng Đối Tượng.* Nhà Xuất Bản Khoa Học & Kỹ Thuật, 2021, p. 401.