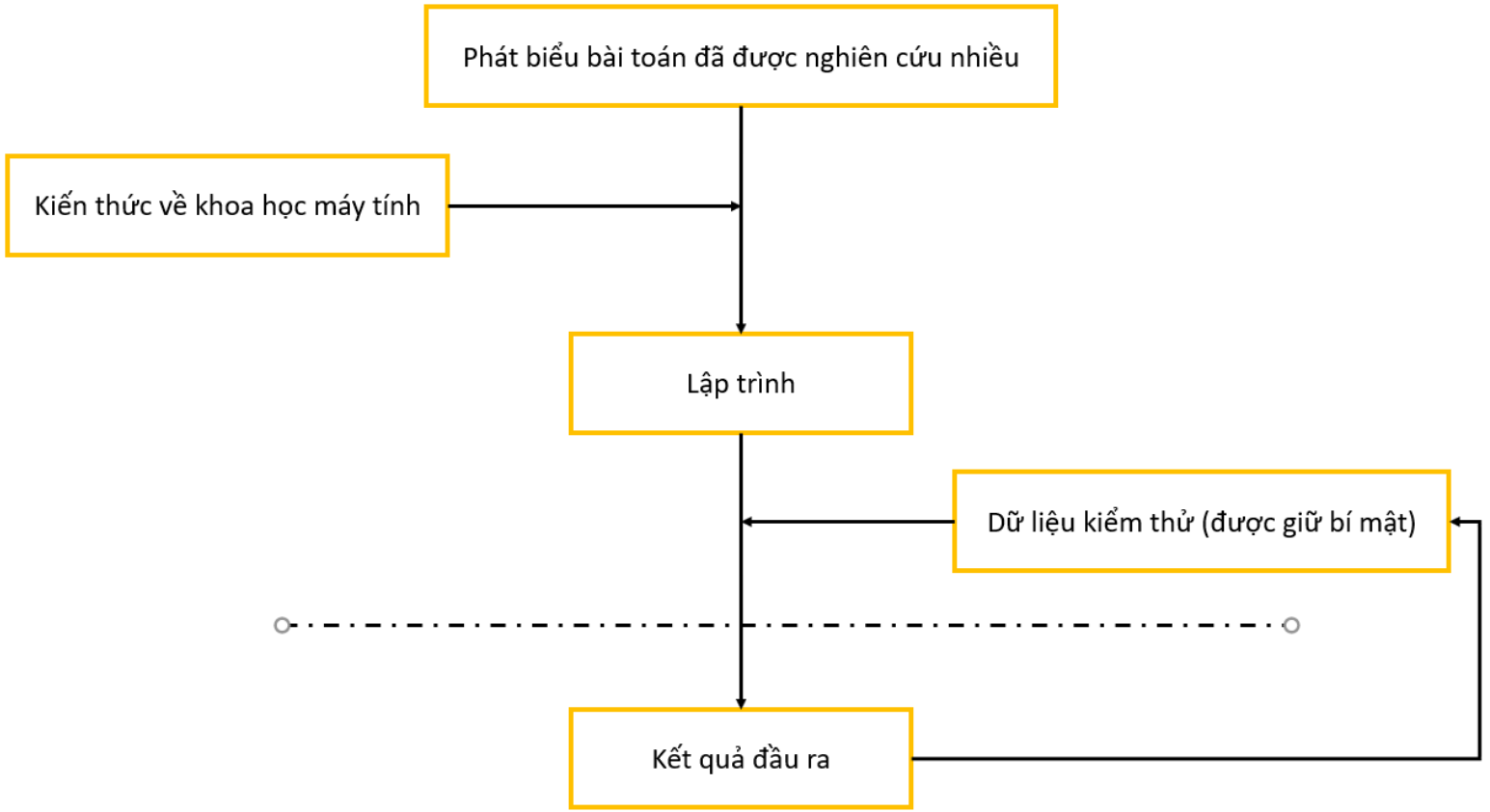
ĐẠI HỌC BÁCH KHOA HÀ NỘI
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

ĐẠI HỌC BÁCH KHOA HÀ NỘI
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

THUẬT TOÁN ỨNG DỤNG

Tư Duy Thuật Toán Và CTDL + Kỹ Năng Lập Trình

Mô hình Bài tập lập trình Thuật toán ứng dụng



Yếu tố chính : giải bài toán đúng nhanh nhất có thể!

Mục tiêu

- Đề bài yêu cầu lập trình giải quyết một bài toán có nội dung ứng dụng thực tế, các vấn đề bao gồm:
  - mô hình hoá bài toán,
  - tìm lời giải hiệu quả sử dụng các thuật toán và cấu trúc dữ liệu,
  - chuyển lời giải thành chương trình và chạy thử nghiệm,
  - làm càng nhanh càng tốt dưới áp lực thời gian và độ chính xác,
  - và phải làm đúng: chương trình không sinh lỗi, kết quả đúng trong thời gian và bộ nhớ hạn chế.

- Mục tiêu của khoá học này là thực hành giải quyết những vấn đề trên.

Phương pháp tiếp cận

- Học những dạng bài phổ biến khác nhau
- Chỉ ra những ứng dụng của các thuật toán và cấu trúc dữ liệu bạn biết từ
  - khóa học cơ bản về các thuật toán
  - khóa học cơ bản về cấu trúc dữ liệu
- Học các dạng thuật toán và cấu trúc dữ liệu phổ biến khác
- Học một số lý thuyết toán/tin học hay dùng
- Thực hành giải bài toán
- Thực hành lập trình
- Thực hành nữa
- .. và thực hành mãi

Tài liệu tham khảo

- Competitive Programming. Steven Halim `http://libgen.io/ads.php?md5=f6f195012783a8b3c8bb7628882a51b7`
- Slides bài giảng Phân tích và thiết kế thuật toán. Nguyễn Đức Nghĩa
- Algorithm design. Jon Kleinberg and Éva Tardos.
- Introduction to Algorithms. T.H. Cormen, C.E. Leiserson, R.L. Rivest, C. Stein.
- Bài giảng Chuyên đề Lê Minh Hoàng
- Competitive Programming Course at Reykjavík University

Các chủ đề dự kiến

Thứ tự	Chủ đề
Buổi 1	Giới thiệu
Buổi 2	Cấu trúc dữ liệu và thư viện
Buổi 3	Thực hành
Buổi 4	Kỹ thuật đệ qui và nhánh cận
Buổi 5	Chia để trị
Buổi 6	Qui hoạch động
Buổi 7	Thực hành
Buổi 8	Qui hoạch động
	Kiểm tra giữa kỳ
Buổi 9	Đồ thị
Buổi 10	Thực hành
Buổi 11	Đồ thị
Buổi 12	Xử lý xâu và thực hành
Buổi 13	Thuật toán tham lam
Buổi 14	Thực hành
Buổi 15	Lớp bài toán NP-đầy đủ

Mẫu đề bài

- Mẫu chuẩn bài toán trong hầu hết các kỳ thi bao gồm:
  - Mô tả bài toán
  - Mô tả định dạng dữ liệu vào
  - Mô tả định dạng kết quả ra
  - Ví dụ Dữ liệu vào/Kết quả ra
  - Giới hạn thời gian theo giây
  - Giới hạn bộ nhớ theo bytes/megabytes
  - Giới hạn kích thước các tham số đầu vào
- Yêu cầu viết chương trình giải bài toán đúng càng nhiều bộ dữ liệu càng tốt
  - Mặc định là dữ liệu vào đúng, không cần kiểm tra tính đúng đắn
  - Chương trình không được chạy quá giới hạn thời gian và giới hạn bộ nhớ

Bài toán ví dụ – ALICEADD

Mô tả bài toán

Alice có a cái kẹo, Bob cho Alice thêm b cái kẹo. Hỏi Alice có tất cả bao nhiêu cái kẹo?

Mô tả dữ liệu vào

- Dòng đầu chứa một số nguyên không âm T là số bộ dữ liệu ($T \leq 10$).
- Mỗi dòng trong số T dòng tiếp theo chứa hai số nguyên không âm a và b cách nhau bởi dấu cách ($a, b \leq 10^{18}$).

Mô tả kết quả ra

Gồm T dòng là kết quả cho T bộ dữ liệu theo thứ tự đầu vào.

Bài toán ví dụ – ALICEADD

Ví dụ dữ liệu vào	Dữ liệu kết quả ra
2 3 5 4 1	8 5

Lời giải ví dụ

```
#include <iostream>
using namespace std;
int main() {
    int T;
    cin >> T;
    for (int i = 0; i < T; i++) {
        int a, b;
        cin >> a >> b;
        cout << a + b << endl;
    }
    return 0;
}
```

Lời giải ví dụ

```cpp
#include <iostream>
using namespace std;
int main() {
    int T;
    cin >> T;
    for (int i = 0; i < T; i++) {
        int a, b;
        cin >> a >> b;
        cout << a + b << endl;
    }
    return 0;
}
```

- Lời giải này có đúng với mọi bộ dữ liệu test không?

Lời giải ví dụ

```cpp
#include <iostream>
using namespace std;
int main() {
    int T;
    cin >> T;
    for (int i = 0; i < T; i++) {
        int a, b;
        cin >> a >> b;
        cout << a + b << endl;
    }
    return 0;
}
```

- Lời giải này có đúng với mọi bộ dữ liệu test không?
- Điều gì xảy ra nếu $a = b = 10^{18}$?

Lời giải ví dụ

```cpp
#include <iostream>
using namespace std;
int main() {
    int T;
    cin >> T;
    for (int i = 0; i < T; i++) {
        int a, b;
        cin >> a >> b;
        cout << a + b << endl;
    }
    return 0;
}
```

- Lời giải này có đúng với mọi bộ dữ liệu test không?
- Điều gì xảy ra nếu $a = b = 10^{18}$? Kết quả phép cộng sẽ bị tràn số lớn.

Lời giải ví dụ

```cpp
#include <iostream>
using namespace std;
int main() {
    int T;
    cin >> T;
    for (int i = 0; i < T; i++) {
        int a, b;
        cin >> a >> b;
        cout << a + b << endl;
    }
    return 0;
}
```

- Lời giải này có đúng với mọi bộ dữ liệu test không? KHÔNG!
- Điều gì xảy ra nếu $a = b = 10^{18}$? Kết quả phép cộng sẽ bị tràn số lớn.

Lời giải ví dụ

- Khi $a = b = 10^{18}$, kết quả phải là 2×10^{18}

Lời giải ví dụ

- Khi $a = b = 10^{18}$, kết quả phải là 2×10^{18}
- Giá trị này quá lớn với biến nguyên 32-bit, nên bị tràn số

Lời giải ví dụ

- Khi $a = b = 10^{18}$, kết quả phải là 2×10^{18}
- Giá trị này quá lớn với biến nguyên 32-bit, nên bị tràn số
- Sử dụng biến nguyên 64-bit sẽ cho lời giải đúng

Lời giải đúng

```cpp
#include <iostream>
using namespace std;
int main() {
    int T;
    cin >> T;
    for (int i = 0; i < T; i++) {
        long long a, b;
        cin >> a >> b;
        cout << a + b << endl;
    }
    return 0;
}
```

Lời giải đúng

```cpp
#include <iostream>
using namespace std;
int main() {
    int T;
    cin >> T;
    for (int i = 0; i < T; i++) {
        long long a, b;
        cin >> a >> b;
        cout << a + b << endl;
    }
    return 0;
}
```

- Lời giải này có đúng không?

Lời giải đúng

```cpp
#include <iostream>
using namespace std;
int main() {
    int T;
    cin >> T;
    for (int i = 0; i < T; i++) {
        long long a, b;
        cin >> a >> b;
        cout << a + b << endl;
    }
    return 0;
}
```

- Lời giải này có đúng không? ĐÚNG!

Lời giải đúng

```cpp
#include <iostream>
using namespace std;
int main() {
    int T;
    cin >> T;
    for (int i = 0; i < T; i++) {
        long long a, b;
        cin >> a >> b;
        cout << a + b << endl;
    }
    return 0;
}
```

- Lời giải này có đúng không? ĐÚNG!
- Làm thế nào nếu giá trị a, b lớn nữa?

Lời giải đúng

```cpp
#include <iostream>
using namespace std;
int main() {
    int T;
    cin >> T;
    for (int i = 0; i < T; i++) {
        long long a, b;
        cin >> a >> b;
        cout << a + b << endl;
    }
    return 0;
}
```

- Lời giải này có đúng không? ĐÚNG!
- Làm thế nào nếu giá trị a, b lớn nữa? XỬ LÝ SỐ LỚN!

Hệ thống chấm điểm trực tuyến tự động

- Một số server phổ biến: `codeforces.com`, `spoj.com`, `ru.kattis.com`, `...`
- Khi nộp bài giải lên hệ thống chấm sẽ phản hồi kết quả ngay
- Hầu hết có thể nộp bài giải với các ngôn ngữ lập trình phổ biến:
  - C/C++
  - Java
  - Python
  - Pascal
  - ...

Hệ thống chấm điểm trực tuyến tự động

- Một số server phổ biến: `codeforces.com`, `spoj.com`, `ru.kattis.com`, `...`
- Khi nộp bài giải lên hệ thống chấm sẽ phản hồi kết quả ngay
- Hầu hết có thể nộp bài giải với các ngôn ngữ lập trình phổ biến:
  - C/C++
  - Java
  - Python
  - Pascal
  - ...

ĐHBKHN sử dụng *server codeforces riêng* để giúp sinh viên thực hành giải bài trực tuyến và sử dụng hệ thống cms kết hợp với hệ thống check trùng code tự động để đánh giá kiểm tra môn học

Một số phản hồi chính

- Các phản hồi từ hệ thống chấm điểm thường không thật chi tiết. Một số phản hồi thường gặp:
  - Kết quả đúng (Accepted)
  - Kết quả sai (Wrong Answer)
  - Chương trình dịch lỗi (Compile Error)
  - Chương trình chạy sinh lỗi (Runtime Error)
  - Quá thời gian cho phép (Time Limit Exceeded)
  - Quá bộ nhớ cho phép (Memory Limit Exceeded)
- Một số server cho phản hồi chi tiết đến từng test.

1 Giới thiệu chung

2 Các kỹ năng cơ bản cần rèn luyện

- A. Kỹ năng đọc đề
- B. Kỹ năng phân loại bài toán
- C. Kỹ năng phân tích thuật toán
- D. Kỹ năng làm chủ ngôn ngữ lập trình
- E. Kỹ năng đặt tên biến
- F. Kỹ năng test chương trình
- G. Kỹ năng gõ nhanh
- I. Kỹ năng thực hành

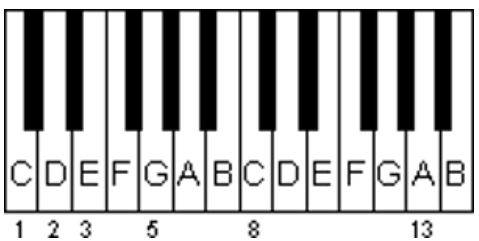
3 Dạng bài toán Ad Hoc

A. Kỹ năng đọc đề

- Kiến thức cơ sở của bài toán (Background): Đây là phần thể hiện Ứng dụng của bài toán
- Các dữ kiện và yêu cầu của bài toán
- Mô tả khuôn dạng dữ liệu vào và ra
- Ví dụ khuôn dạng vào ra: thường chỉ là những test đơn giản
- Các hạn chế (kích thước dữ liệu kiểm thử, thời gian, bộ nhớ) cho các test của bài toán

Phần Background có quan trọng không?

- KHÔNG?
  - Ví dụ: *"Hãy lập trình sắp xếp n số thực. Hạn chế bộ nhớ: 3MB. Hạn chế thời gian: 1 giây. Hạn chế kích thước đầu vào $n \leq 10^6$. Hạn chế giá trị số thực trong 4 byte với định dạng đầu vào có chính xác hai chữ số sau dấu phẩy động"*.
  - Quá ngắn gọn và dễ hiểu!
- Tin học: BUỘC PHẢI CÓ!!!
  - Ví dụ: *Bản vanxơ Fibonacci là một bản nhạc mà giai điệu của nó bắt nguồn từ một trong những dãy số nổi tiếng nhất trong Lý thuyết số - dãy số Fibonacci. Hai số đầu tiên của dãy là số 1 và số 2, các số tiếp theo được xác định bằng tổng của hai số liên tiếp ngay trước nó trong dãy. Bản vanxơ Fibonacci thu được bằng việc chuyển dãy số Fibonacci thành dãy các nốt nhạc theo qui tắc chuyển một số nguyên dương thành nốt nhạc sau đây:*



[Ấn vào đây để nghe bản nhạc]

  - Tạo hứng cho người đọc giải bài
  - Một chủ đề ứng dụng của KHMT

B. Kỹ năng phân loại bài toán

- Thực hành phân loại nhanh các dạng bài toán
- Có thể là kết hợp của nhiều dạng khác nhau
- Một số dạng bài hay gặp:

Loại	Loại hẹp
Ad Hoc	Trực tiếp, Mô phỏng
Duyệt toàn bộ	Lặp, Quay lui, Nhánh cận
Chia để trị	Cổ điển, Cải tiến
Giảm để trị	Cổ điển, Cải tiến
Tham lam	Cổ điển, Cải tiến
Quy hoạch động	Cổ điển, Cải tiến
Đồ thị	Cổ điển, Cải tiến
Đồ thị đặc biệt	Cổ điển, Cải tiến
Toán học	Cổ điển, Cải tiến
Xử lý xâu	Cổ điển, Cải tiến
Tính toán hình học	Cổ điển, Cải tiến
Một số loại thuật toán đặc thù	

C. Kỹ năng phân tích thuật toán

- Lời giải bài toán phải đủ nhanh và không sử dụng quá nhiều bộ nhớ
- Lời giải nên càng đơn giản càng tốt
- Sử dụng phương pháp Phân Tích Thuật Toán để xác định xem lời giải đưa ra có thỏa mãn giới hạn thời gian và bộ nhớ không
- Thông thường tính: 5×10^8 phép tính trong một giây
- Ví dụ cần sắp xếp $n \leq 10^6$ số nguyên chạy trong 1 giây
  - Liệu có thể sử dụng sắp xếp nổi bọt, dễ cài đặt, và có độ phức tạp $\mathcal{O}(n^2)$ được không?
  - Sử dụng một thuật toán sắp xếp nhanh, khó cài đặt hơn, và có độ phức tạp $\mathcal{O}(n \log n)$?
- Nếu sắp xếp $n \leq 10^3$ số nguyên chạy trong 1 giây
  - Bây giờ có thể sử dụng sắp xếp nổi bọt được không?
- Luôn nhớ hãy sử dụng giải pháp đơn giản nhất thỏa mãn giới hạn thời gian và bộ nhớ

- Hãy thực hành cách ước lượng xấp xỉ trong đầu
- Mẹo:
  - $2^{10} \approx 10^3$
  - $\log_2 10 \approx 3$

- Trường hợp tìm ra lời giải mà bạn không chắc là nó đúng thì:
  - Hãy tìm cách chứng minh nó!
  - Thử tính đúng sai với các bộ kiểm thử có kích thước nhỏ
  - Ngay cả khi không tìm được cách chứng minh hoặc phản bác nó thì điều thu được là hiểu sâu thêm bài toán

n	Độ phức tạp sát nhất	Ví dụ
≤ 10	$\mathcal{O}(n!), \mathcal{O}(n^6)$	Liệt kê hoán vị
≤ 15	$\mathcal{O}(2^n \times n^2)$	QHĐ+ Kỹ thuật bitmask cho bài TSP
≤ 20	$\mathcal{O}(2^n), O(n^5)$	Liệt kê nhị phân, QHĐ 4-5 chiều
≤ 50	$\mathcal{O}(n^4)$	QHĐ 3-4 chiều, Liệt kê tổ hợp $C_n^{k=4}$
$\leq 10^2$	$\mathcal{O}(n^3)$	Floyd Warshall
$\leq 10^3$	$\mathcal{O}(n^2)$	Sắp xếp Nổi bọt/Chọn/Chèn
$\leq 10^6$	$\mathcal{O}(n \log_2 n)$	Sắp xếp Trộn, Cây phân đoạn (Segment/Interval)
	$\mathcal{O}(n), \mathcal{O}(\log_2 n), O(1)$	Thường kích thước đầu vào bài toán $n \leq 10^6$

D. Kỹ năng làm chủ ngôn ngữ lập trình

- Hãy thành thạo ngôn ngữ lập trình như lòng bàn tay
- Bao gồm cả các thư viện có sẵn
  - Thư viện STL của C++ (Standard Template Library)
  - Thư viện của Java (Java Class Library)
- Nếu đã có sẵn trong thư viện chuẩn thì không cần phải lập trình lại: cài đặt nhanh, không có bug
- Hạn chế : khả năng tùy biến của thư viện không linh hoạt. Rất nhiều phần kỹ thuật xử lý thuật toán không thể gọi trực tiếp thư viện mà phải tùy biến đi hoặc phải cài đặt lại

E. Kỹ năng đặt tên biến

- Tên hàm viết hoa chữ cái đầu. Ví dụ: `void ReadInp()`
- Tên hằng số viết in hoa. Ví dụ: `const int MAX=100000;`
- Tên mảng bắt đầu bằng chữ cái đầu của kiểu giá trị: `int` $\rightarrow$ `i`, tiếp theo là viết hoa chữ cái đầu. Ví dụ: `int iCost[MAX], bool bMark[1001], string sLine[100];`
- Tên biến đơn viết thường. Ví dụ: `int i, u, sum, res, ans;`
- Tên biến nên càng giống với tên được cho trong bài toán càng tốt
- ...

F. Kỹ năng test/debug chương trình

- Phải test để chắc chắn kết quả bài giải là đúng và thỏa mãn giới hạn thời gian và bộ nhớ, hoặc ít ra là biết lời giải chưa đúng
- Cố gắng phản biện bài giải bằng cách tìm ra phản ví dụ (một dữ liệu vào mà bài giải trả kết quả ra sai, hoặc mất quá nhiều thời gian để tìm ra kết quả)
- Test các bộ dữ liệu biên và dữ liệu lớn, ...
- Viết một thuật toán trực tiếp đơn giản để kiểm tra kết quả chương trình của mình có đúng không với những trường hợp kích thước đầu vào nhỏ

G. Kỹ năng gõ nhanh

- Hãy trở thành thợ gõ nhanh và chính xác hơn
- Đừng để việc gõ lời giải là hạn chế cho việc giải bài
- Khi đánh máy không nhìn vào bàn phím, mắt nhìn vào màn hình kiểm tra luôn tính đúng đắn của việc gõ, trong lúc đó đầu vẫn có thể suy nghĩ song song các vấn đề tiếp theo

- Ví dụ: sử dụng TypeRacer là một cách luyện tập hiệu quả và thú vị: http://play.typeracer.com/

I. Thực hành, thực hành nữa và thực hành mãi

- Càng thực hành nhiều thì càng hoàn thiện các kỹ năng giải bài và lập trình
- Rất nhiều các trang chấm bài trực tuyến giúp bạn giải các bài toán của những kỳ thi trong quá khứ
- Một số trang giải bài trực tuyến thường xuyên tổ chức các cuộc thi đấu lập trình: Codeforces, TopCoder, Codechef, Kattis, ...

Lập trình viên thi đấu cũng như những vận động viên thể thao, cần phải rèn luyện thường xuyên để giữ được cảm hứng và phát triển các kỹ năng lập trình giải bài!

Các bài toán Ad Hoc

Loại bài toán Ad Hoc

- Là dạng bài toán đơn giản nhất
- Thường làm đúng như mô tả bài toán yêu cầu
- Trực tiếp hoặc Mô phỏng
- Giới hạn thời gian thường không quan trọng
- Đôi khi mô tả dài dòng khó hiểu
- Đôi khi một số test biên lừa
- Một số bài toán phức hợp có thể khó lập trình

Bài toán: Cắt giảm cửa hàng

Đại dịch COVID19 trên toàn thế giới khiến người dân rất hạn chế ra khỏi nhà, điều này đã đẩy rất nhiều công ty phải phá sản và rất nhiều công ty khác phải cắt giảm bớt các hệ thống cửa hàng, văn phòng, sa thải nhân viên, giảm lương thưởng,...

Công ty XTEC cũng không phải ngoại lệ. Họ có 4 cửa hàng và quyết định đóng cửa tối đa 2 trong số đó có lợi nhuận âm thấp nhất năm 2019.

Yêu cầu: Cho trước lợi nhuận năm 2019 của 4 cửa hàng, hãy đưa ra tổng lợi nhuận âm của những cửa hàng phải đóng cửa.

Bài toán: Cắt giảm cửa hàng

Dữ liệu vào

- Dòng đầu tiên chưa một số nguyên T ($T < 20$) là số lượng trường hợp test.
- Mỗi dòng trong số T dòng sau chứa 4 số nguyên phân biệt biểu diễn lợi nhuận năm 2019 của 4 cửa hàng. Tất cả các số nguyên ở đây nằm trong khoảng $[-10000, 10000]$.

Kết quả ra

Mỗi test ghi trên một dòng một số duy nhất là tổng lợi nhuận âm của các cửa hàng phải cắt bỏ.

Bài toán: Cắt giảm cửa hàng

Ví dụ dữ liệu vào	Ví dụ kết quả ra
3 -1000 2000 3000 -4000 3000 -2500 1500 100 -1500 -1200 -1800 -1900	-5000 -2500 -3700

Lời giải bài toán: Cắt giảm cửa hàng

```cpp
#include<bits/stdc++.h>
using namespace std;
int main() {
    ios_base::sync_with_stdio(0);
    cin.tie(0);cout.tie(0);

    int iProfit[4];
    int T; cin >> T;
    for (int i = 0; i < T; i++) {
        cin >> iProfit[1] >> iProfit[2];
        cin >> iProfit[3] >> iProfit[4];

        sort(iProfit, iProfit + 4);
        int sum=0;
        if (iProfit[1]<0) sum+=iProfit[1];
        if (iProfit[2]<0) sum+=iProfit[2];
        cout << sum << endl;
    }
    return 0;
}
```

Bài toán: Gõ SMS

Điện thoại di động (ĐTDĐ) trở nên một phần không thể thiếu trong cuộc sống hiện đại. Ngoài việc gọi, ĐTDĐ có thể gửi tin nhắn mà người ta quen gọi là SMS. Không như bàn phím máy tính, đa phần ĐTDĐ cổ điển hạn chế số phím. Để có thể gõ được tất cả các ký tự trong bảng chữ cái, nhiều ký tự sẽ được hiển thị trên cùng một phím. Vì vậy, để gõ một số ký tự, một phím sẽ phải được ấn liên tục đến khi ký tự cần tìm hiển thị trên màn hình.

Cho một đoạn văn bản, hãy tính số lần gõ phím để hiển thị được đoạn văn bản.

Bài toán: Gõ SMS

Bài toán giả thiết rằng các phím được sắp xếp như sau.

	abc	def
ghi	jkl	mno
pqrs	tuv	wxyz
	<SP>	

Trong bảng trên mỗi ô biểu diễn một phím. <SP> biểu diễn phím space. Để hiển thị ký tự 'a' thì sẽ phải bấm phím tương ứng 1 lần, nhưng để hiển thị ký tự 'b' của cùng phím đó thì sẽ phải bấm liên tục 2 lần và đối với phím 'c' là 3 lần. Tương tự, bấm 1 lần cho 'd', hai lần cho 'e' và 3 lần cho 'f'. Các ký tự khác cũng được làm tương tự. Lưu ý là để ra 1 khoảng trống thì cần bấm 1 lần phím space.

Bài toán: Gõ SMS

Dữ liệu vào

Dòng đầu tiên là một số nguyên T là số lượng trường hợp test. T dòng tiếp theo mỗi dòng chỉ chứa các khoảng trống và các ký tự in thường. Mỗi dòng chứa ít nhất 1 và tối đa 100 ký tự.

Kết quả ra

Mỗi trường hợp test đầu vào tương ứng với một dòng ở kết quả ra. Mỗi dòng bắt đầu bởi thứ tự trường hợp test và sau đó là một số biểu thị số lần bấm phím cho ra văn bản tương ứng. Xem ví dụ kết quả ra để thấy định dạng chuẩn xác.

Bài toán: Gõ SMS

Ví dụ dữ liệu vào	Ví dụ kết quả ra
2 welcome to ulab good luck and have fun	Case #1: 29 Case #2: 41

Lời giải bài toán: Gõ SMS

```cpp
#include<bits/stdc++.h>
using namespace std;
string sKey[12] = {
  "",     "abc", "def",
  "ghi",  "jkl", "mno",
  "pqrs", "tuv", "wxyz",
  "",     " ",   ""
};
int main() {
  ios_base::sync_with_stdio(0);
  cin.tie(0);cout.tie(0);
  int T; cin >>T;
  for (int t = 0; t < T; t++) {
    // Mỗi trường hợp Test được thực hiện ở đây
  }
  return 0;
}
```

Lời giải bài toán: Gõ SMS

```
1   // Mỗi trường hợp Test được thực hiện ở đây
2   string sLine;
3   getline(cin, sLine);
4   int res = 0;
5   for (int i = 0; i < sLine.size(); i++) {
6     int cur;
7     for (int j = 0; j < 12; j++) {
8       for (int k = 0; k < sKey[j].size(); k++) {
9           if (sLine[i] == sKey[j][k]) {
10              cur = k + 1;
11          }
12      }
13    }
14    res += cur;
15  }
16  cout<<"Case #<<"<<T + 1<<": "<<res);
```

BÀI TẬP THỰC HÀNH

- ALICEADD
- SUBSEQMAX

VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG
SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY

Thank you for your attentions!

soict.hust.edu.vn/ fb.com/groups/soict