

		3244242465641232 2222229999992323211111	1244242465643232 1222229999992323211112
W9A4	<p>Cho xâu a gồm các chữ cái thường và in hoa và xâu b chỉ gồm các chữ cái in hoa. Ta có thể biến đổi xâu a theo 2 bước như sau:</p> <ul style="list-style-type: none"> - Chuyển một số chữ cái thường trong xâu a thành chữ cái in hoa (có thể không biến đổi chữ cái nào) - Xóa hết tất cả các chữ cái thường trong xâu a. <p>Cho 2 xâu a,b, xác định xem có cách biến đổi từ xâu a sang xâu b không. Ghi ra "YES" nếu có và "NO" nếu ngược lại.</p> <p>Đầu vào: Dòng đầu tiên chứa 1 số nguyên Q là số lượng câu hỏi. Tương ứng với mỗi câu hỏi, đưa ra 2 xâu a,b trên 2 dòng khác nhau. ($Q \leq 10, a , b \leq 1000$)</p> <p>Đầu ra: Với mỗi câu hỏi, ghi ra kết quả tương ứng trên một dòng.</p>	<p>1 daBcd ABC</p> <p>2 daBcd ABCE abCDNCGabcd CDNCGABAB</p> <p>1 abcHJS ACHJS</p>	<p>YES</p> <p>NO NO</p> <p>YES</p>
W9A5	<p>Cho hai mảng số nguyên list1 và list2, viết hàm kiểm tra xem list1 có thuộc list2 hay không (có phải là dãy con liên tiếp của list2 hay không)?</p>	<p>1 2 3 4 2 3 1 2 3 4 5 6</p> <p>1 2 3 4 2 3 1 2 3 5 4 6 7 8</p> <p>1 2 3 4 2 3 1 2 3 4 5 6 1 2 3 4</p>	<p>YES NO YES</p>
W9A6	<p>[Hoán vị tiếp theo] Cho một dãy số gồm n chữ số là một hoán vị của dãy 1,2,...,n. Viết chương trình in ra hoán vị ở vị trí tiếp theo (danh sách các hoán vị được sắp xếp theo thứ tự từ điển) của hoán vị đã cho.</p> <p>Đầu vào từ bàn phím gồm 2 dòng:</p> <ul style="list-style-type: none"> - Dòng đầu tiên chứa số nguyên n - Dòng tiếp theo chứa n số nguyên là một hoán vị của dãy 1,2,...,n ($n \leq 105$) <p>Đầu ra: In ra màn hình hoán vị tiếp theo của hoán vị đã cho.</p>	<p>3 1 2 3</p> <p>4 1 4 3 2</p> <p>5 1 2 3 5 4</p> <p>6 4 5 6 3 2 1</p>	<p>1 3 2</p> <p>2 1 3 4</p> <p>1 2 4 3 5</p> <p>4 6 1 2 3 5</p> <p>1 2 3</p>

		3 3 2 1	
W9A7	<p>[CanPlaceFlowers] Giả sử bạn là nhân viên của một công ty cây xanh, được giao cho chăm sóc một luống đất. Luống đất đó được chia thành nhiều ô đất nhỏ có kích cỡ bằng nhau. Trước khi được giao cho bạn chăm sóc, trên luống hoa đã được trồng sẵn một số cây xanh tán rộng. Một ngày đầu xuân, công ty gửi đến cho bạn một số cây hoa và yêu cầu bạn trồng vào những vị trí còn trống trong luống hoa. Đây là những cây hoa hướng sáng và chúng không thể sống được dưới bóng râm của cây khác, đặc biệt là cây tán rộng. Vì vậy, bạn không thể trồng chúng tại ô bên cạnh những ô đã được trồng cây xanh tán rộng. Và nếu trồng những cây hoa bên cạnh nhau, chúng sẽ tranh giành phân bón và nước và sau cùng, cả hai cây sẽ cùng chết.</p> <p>Viết hàm <code>canPlaceFlowers(flowerbed, k)</code> kiểm tra xem có thể trồng hết số hoa mới được cấp vào trong luống đất của bạn hay không.</p> <p>Hàm nhận đầu vào là một danh sách (list) <code>flowerbed</code>, đánh dấu những vị trí trong luống đất đã được trồng cây xanh trước đó (giá trị 1 ứng với vị trí cây đã được trồng và 0 ứng với vị trí đất trống) và số lượng hoa được phát thêm <code>k</code> (int). Hàm trả về <code>True</code> nếu có thể trồng hết toàn bộ <code>k</code> cây hoa vào luống đất và <code>False</code> trong trường hợp ngược lại.</p>	<p>1 0 0 0 1 1</p> <p>1 0 0 0 1 2</p> <p>0 0 0 0 0 3</p> <p>1 0 1 0 1 1</p> <p>0 1</p>	<p>True</p> <p>False</p> <p>True</p> <p>False</p> <p>True</p>
W9A8	<p>[Phát hiện đường đi lặp lại của Rô-bốt] Ban đầu, một con rô-bốt ở vị trí (0, 0) trên một lưới 2D. Cho một chuỗi <code>`moves`</code> chứa các thao tác di chuyển của rô-bốt.</p> <p>Các bước di chuyển được biểu diễn bởi các ký tự: - 'R' (Phải) - 'L' (Trái) - 'U' (Lên) - 'D' (Xuống)</p> <p>Viết hàm <code>`def is_path_crossing(moves)`</code>: Hàm nhận đầu vào là một chuỗi <code>`moves`</code>. Hàm trả về <code>`True`</code> nếu trong quá trình di chuyển, rô-bốt đi vào một</p>	<p>RU</p> <p>UD</p> <p>RDRU</p> <p>RUULL</p> <p>RURDL</p>	<p>False</p> <p>True</p> <p>True</p> <p>False</p> <p>True</p>

	<p>ô mà nó đã từng đi qua trước đó. Hàm trả về `False` nếu rô-bốt không bao giờ đi lặp lại (luôn đi vào ô mới).</p> <p>Lưu ý: Vị trí ban đầu (0, 0) được tính là vị trí đầu tiên đã đi qua.</p>		
W9A9	<p>[HARD - Tìm trung vị của 5 số]</p> <p>Viết hàm `def Median(nums):` Hàm nhận đầu vào là một danh sách (list) `nums` được đảm bảo có đúng 5 số nguyên.</p> <p>Yêu cầu: Tìm và trả về số trung vị (số lớn thứ ba) của 5 số này.</p> <p>Ràng buộc: 1. Chỉ được sử dụng tối đa 6 phép so sánh (ví dụ: <, >, <=, >=). 2. KHÔNG ĐƯỢC DÙNG bất kỳ hàm sắp xếp nào (như `sorted()`, `list.sort()`).</p> <p>Chú ý: Bạn chỉ cần viết hàm `Median` theo yêu cầu. Phần đọc input và gọi hàm (hàm main) đã có sẵn.</p>	<p>1 2 3 4 5</p> <p>5 4 3 2 1</p> <p>1 5 3 4 2</p> <p>1 10 3 8 5</p> <p>1 4 3 8 5</p>	<p>4</p> <p>3</p> <p>3</p> <p>10</p> <p>5</p>
W9A10	<p>[Tìm bán kính lò sưởi nhỏ nhất]</p> <p>Mùa đông đang đến! Bạn được cung cấp vị trí của các ngôi nhà và các lò sưởi trên một đường thẳng. Nhiệm vụ của bạn là tìm bán kính r nhỏ nhất cho các lò sưởi sao cho mọi ngôi nhà đều nằm trong phạm vi sưởi ấm của ít nhất một lò sưởi.</p> <p>Tất cả các lò sưởi đều có cùng một bán kính r. Một ngôi nhà tại vị trí h được sưởi ấm bởi một lò sưởi tại vị trí t nếu $h - t \leq r$.</p> <p>Viết hàm `def findRadius(houses, heaters):` Hàm nhận đầu vào là hai danh sách (list) các số nguyên: - `houses`: chứa vị trí của các ngôi nhà. - `heaters`: chứa vị trí của các lò sưởi. Hàm trả về một số nguyên là bán kính r nhỏ nhất cần thiết.</p> <p>Lưu ý:</p>	<p>1 2 3</p> <p>2</p> <p>3 91 41234 4123 4 12348 23323 4192 32 123 4</p> <p>21324 12 31</p> <p>0</p> <p>104231</p> <p>43 123 412123 4 2 32 23 90</p> <p>0</p> <p>12312354</p> <p>15042459 746409159 135838565 89901416</p> <p>6489069 82463372 45389508 271597517</p> <p>713600645 831821015 998718780 432328078</p> <p>693445585 58632669 311066704 905436329</p> <p>181822818 782411037 520556150 404533088</p> <p>880173822 562148154 366934571 653597300</p>	<p>1</p> <p>41234</p> <p>104108</p> <p>12312354</p> <p>67746114</p>

	<ul style="list-style-type: none"> - Số lượng nhà và lò sưởi là một số không âm và không vượt quá 25000. - Vị trí của các ngôi nhà và lò sưởi là một số không âm và không vượt quá 10^9. 	119430172 965103639 21085028 774880949 203851403 596644917	
--	---	---	--