**Jugs**

1997 ACM South Central USA

Programming Contest

Problem #7: **Jugs**

In the movie "Die Hard 3", Bruce Willis and Samuel L. Jackson were confronted with the following puzzle. They were given a 3-gallon jug and a 5-gallon jug and were asked to fill the 5-gallon jug with exactly 4 gallons. This problem generalizes that puzzle.

You have two jugs, A and B, and an infinite supply of water. There are three types of actions that you can use: (1) you can fill a jug, (2) you can empty a jug, and (3) you can pour from one jug to the other. Pouring from one jug to the other stops when the first jug is empty or the second jug is full, whichever comes first. For example, if A has 5 gallons and B has 6 gallons and a capacity of 8, then pouring from A to B leaves B full and 3 gallons in A.

A problem is defined by a triple (Ca, Cb, N), where Ca and Cb are the capacities of the jugs A and B, respectively, and N is the goal. A solution is a sequence of steps that leaves exactly N gallons in jug B. The possible steps are

fill A   
fill B   
empty A   
empty B   
pour A B   
pour B A   
success

where "pour A B" means "pour the contents of jug A into jug B", and "success" means that the goal has been accomplished.

You may assume that the input you are given does have a solution.

**Input Format**

Input to your program consists of a series of input lines each defining one puzzle. Input for each puzzle is a single line of three positive integers: Ca, Cb, and N, where Ca and Cb are the capacities of jugs A and B, and N is the goal. You can assume 0 < Ca <= Cb and N <= Cb <=1000 and that A and B are relatively prime to one another.

**Required Output Format**

Output from your program will consist of lines of instructions which will result in the B jug containing exactly N gallons of water. The last line of output for each puzzle should be the line "success".

|  |  |  |  |
| --- | --- | --- | --- |
| **Input** jugs1.txt  3 5 4 |  | **Input** jugs2.txt  3 5 4 5 7 3  11 13 9 27 59 17 19 37 1 67 91 19 1 1 1 |  |
| **Output** jugs1.out  fill B pour B A empty A pour B A fill B pour B A success |  | **Output** jugs2.out  fill B  pour B A  empty A  pour B A  fill B  pour B A  success  fill A  pour A B  fill A  pour A B  empty B  pour A B  success  fill A  pour A B  fill A  pour A B  empty B  pour A B  success  fill A  pour A B  fill A  pour A B  fill A  pour A B | empty B  pour A B  fill A  pour A B  fill A  pour A B  empty B  pour A B  success  fill A  pour A B  fill A  pour A B  empty B  pour A B  success  fill A  pour A B  fill A  pour A B  empty B  pour A B  fill A  pour A B  empty B  pour A B  success  fill B  success |

**Comment**

The solution views the set of jug content pairs as vertices in a graph, with an edge from one pair/vertex to another if and only if one of the fill/empty/pour actions gets you from one configuration to the other.  After the graph is built, the solution is just a breadth-first search of the graph.  Instead of returning a list of reachable vertices, you build and return a string of actions. (A depth-first search would also find the solution, and perhaps be simpler with recursion, but the breadth-first search gives the shortest sequence of jug actions to achieve the desired result.)

**Help**

1. Which algorithm will solve this problem?
2. Draw the path through the Jugs graph to reach (3, 5, 4).
3. What does the Jug class need to do?