Water and Jug Problem

You are given two jugs with capacities *x* and *y* litres. There is an infinite amount of water supply available. You need to determine whether it is possible to measure exactly *z* litres using these two jugs.

If *z* liters of water is measurable, you must have *z* liters of water contained within **one or both buckets** by the end.

Operations allowed:

- Fill any of the jugs completely with water.
- Empty any of the jugs.
- Pour water from one jug into another till the other jug is completely full or the first jug itself is empty.

Example 1: (From the famous *"Die Hard"* example)

```
Input: x = 3, y = 5, z = 4
Output: True
```

Example 2:

```
Input: x = 2, y = 6, z = 5
Output: False
```

Credits:
Special thanks to @vinod23 for adding this problem and creating all test cases.

Solution 1

only thing we should proof is this:

```
if `x` and `y` are coprime,
then we can and only can reach every integer `z` in `[0, x + y]`. (1)
```

then for a GCD `g`, from `gx` and `gy`, we can and only can reach every `z` in `{i * g | i in [0, x + y] }`

now, let's see how to proof (1). let x be the less one, and y the greater one. then fill the two jug to full, we have `x` and `y` water each and `x + y` water in total. then we pour out x water each time until we can't.

now we have these different z:

```
y + x,  y,  y - x,  y - 2x, ... , y % x
```

finally we have `y % x` water left, we pour it into the x jug, then fill the y jug to full. now the two jugs have `y % x` and `y` water separately, and `y + y % x` water in total. then we pour from y jug into x jug until the x jug is full, afterwards do the same thing like before, to pour out x water each time until we can't.

finally we get `(y + y % x) % x = (y % x + y % x) % x = (2y) % x` water left.

now we have these different z:

```
y + y % x,   y + y % x - x,  y + y % x - 2x, ... ,  (2y) % x
```

do this x times, we get z:

```
y + (2y) % x,  y + (2y) % x - x, y + (2y) % x - 2x, ..., (3y) % x
:
:
:
y + ((x-1)y) % x,  y + ((x-1)y) % x - x,  y + ((x-1)y) % x - 2x, ... ,  (xy) % x
```

then you see `(xy) % x = 0`, and

```
y % x, (2y) % x, (3y) % x, ... , ((x-1)y) % x just are 1, 2, 3, 4, 5, ... , x - 1. (2)
```

proof for (2): modulo x could get x - 1 different results at most exclusive 0, that's 1,2,3,...,x-1. we have x - 1 expressions, suppose there is two same, let a != b in [1, x-1] and `(ay) % x = (by) % x`, then we get `((a - b)y) % x = 0`, then `((a - b) % x) * (y % x) = 0`, `(a - b) % x = 0`. for `1 <= a, b <= x - 1`, so we get `a = b`. it's impossible.

so we can get any z in [0, x + y] water for coprime x and y.

code:

```
   bool canMeasureWater(int x, int y, int z) {
    return z == 0 || z <= (long long)x + y && z % __gcd(x, y) == 0;
}
```

p.s. the testcases are too weak. you may get overflow without long long, say 2147483647, 234524287, 1. some code get AC, while runs wrong on this 4,6,8.

written by lishq991 original link here

Solution 2

This is a pure Math problem. We need the knowledge of number theory to cover the proof and solution. No idea why microsoft uses this problem in real interview.

The basic idea is to use the property of Bézout's identity and check if z is a multiple of GCD(x, y)

Quote from wiki:

> Bézout's identity (also called Bézout's lemma) is a theorem in the elementary theory of numbers:
>
> let a and b be nonzero integers and let d be their greatest common divisor. Then there exist integers x and y such that ax+by=d
>
> In addition, the greatest common divisor d is the smallest positive integer that can be written as ax + by
>
> every integer of the form ax + by is a multiple of the greatest common divisor d.

If a or b is negative this means we are emptying a jug of x or y gallons respectively.

Similarly if a or b is positive this means we are filling a jug of x or y gallons respectively.

x = 4, y = 6, z = 8.

GCD(4, 6) = 2

8 is multiple of 2

so this input is valid and we have:

-1 * 4 + 6 * 2 = 8

In this case, there is a solution obtained by filling the 6 gallon jug twice and emptying the 4 gallon jug once. (Solution. Fill the 6 gallon jug and empty 4 gallons to the 4 gallon jug. Empty the 4 gallon jug. Now empty the remaining two gallons from the 6 gallon jug to the 4 gallon jug. Next refill the 6 gallon jug. This gives 8 gallons in the end)

See wiki:

Bézout's identity

and comments in the code

```java
public boolean canMeasureWater(int x, int y, int z) {
    //limit brought by the statement that water is finallly in one or both buckets
    if(x + y < z) return false;
    //case x or y is zero
    if( x == z || y == z || x + y == z ) return true;

    //get GCD, then we can use the property of Bézout's identity
    return z%GCD(x, y) == 0;
}

public int GCD(int a, int b){
    while(b != 0 ){
        int temp = b;
        b = a%b;
        a = temp;
    }
    return a;
}
```

written by hpplayer original link here

Solution 3

Why the test case 1 2 5 return false? Because I think 5 = 1 + 2 + 2.

written by fangdanzai original link here