

# Measure one litre using two vessels and infinite water supply

There are two vessels of capacities 'a' and 'b' respectively. We have infinite water supply. Give an efficient algorithm to make exactly 1 litre of water in one of the vessels. You can throw all the water from any vessel any point of time. Assume that 'a' and 'b' are **Coprimes**.

Following are the steps:

Let V1 be the vessel of capacity 'a' and V2 be the vessel of capacity 'b' and 'a' is smaller than 'b'.

1) Do following while the amount of water in V1 is not 1.

....a) If V1 is empty, then completely fill V1

....b) Transfer water from V1 to V2. If V2 becomes full, then keep the remaining water in V1 and empty V2

2) V1 will have 1 litre after termination of loop in step 1. Return.

Following is C++ implementation of the above algorithm.

```
/* Sample run of the Algo for V1 with capacity 3 and V2 1
1. Fill V1: V1 = 3, V2 = 0
2. Transfer from V1 to V2, and fill V1: V1 = 3, V2 = 3
2. Transfer from V1 to V2, and fill V1: V1 = 3, V2 = 6
3. Transfer from V1 to V2, and empty V2: V1 = 2, V2 = 0
4. Transfer from V1 to V2, and fill V1: V1 = 3, V2 = 2
5. Transfer from V1 to V2, and fill V1: V1 = 3, V2 = 5
6. Transfer from V1 to V2, and empty V2: V1 = 1, V2 = 0
7. Stop as V1 now contains 1 litre.
```

Note that V2 was made empty in steps 3 and 6 because it 1

```
#include <iostream>
using namespace std;
```

```
// A utility function to get GCD of two numbers
int gcd(int a, int b) { return b? gcd(b, a % b) : a; }
```

```
// Class to represent a Vessel
class Vessel
{
```

```

// A vessel has capacity, and current amount of water
int capacity, current;
public:
// Constructor: initializes capacity as given, and c
Vessel(int capacity) { this->capacity = capacity; cu

// The main function to fill one litre in this vessel. C
// must be greater than this vessel and two capacitio
void makeOneLitre(Vessel &V2);

// Fills vessel with given amount and returns the am
// transferred to it. If the vessel becomes full, the
// is made empty.
int transfer(int amount);
};

```

```

// The main function to fill one litre in this vessel. C
// of V2 must be greater than this vessel and two capacitio
// must be coprime
void Vessel:: makeOneLitre(Vessel &V2)
{
// solution exists iff a and b are co-prime
if (gcd(capacity, V2.capacity) != 1)
    return;

while (current != 1)
{
// fill A (smaller vessel)
if (current == 0)
    current = capacity;

cout << "Vessel 1: " << current << "    Vessel 2: "
    << V2.current << endl;

// Transfer water from V1 to V2 and reduce curren
// the amount equal to transferred water
current = current - V2.transfer(current);
}

// Finally, there will be 1 litre in vessel 1
cout << "Vessel 1: " << current << "    Vessel 2: "
    << V2.current << endl;
}

```

```

// Fills vessel with given amount and returns the amount
// transferred to it. If the vessel becomes full, then tl

```

```
// is made empty
int Vessel::transfer(int amount)
{
    // If the vessel can accommodate the given amount
    if (current + amount < capacity)
    {
        current += amount;
        return amount;
    }

    // If the vessel cannot accommodate the given amount
    // store the amount of water transferred
    int transferred = capacity - current;

    // Since the vessel becomes full, make the vessel
    // empty so that it can be filled again
    current = 0;


    return transferred;
}

// Driver program to test above function
int main()
{
    int a = 3, b = 7; // a must be smaller than b

    // Create two vessels of capacities a and b
    Vessel V1(a), V2(b);

    // Get 1 litre in first vessel
    V1.makeOneLitre(V2);

    return 0;
}
```



Output:

```
Vessel 1: 3    Vessel 2: 0
Vessel 1: 3    Vessel 2: 3
Vessel 1: 3    Vessel 2: 6
Vessel 1: 2    Vessel 2: 0
Vessel 1: 3    Vessel 2: 2
Vessel 1: 3    Vessel 2: 5
```

Vessel 1: 1    Vessel 2: 0

## How does this work?

To prove that the algorithm works, we need to prove that after certain number of iterations in the while loop, we will get 1 litre in V1.

Let 'a' be the capacity of vessel V1 and 'b' be the capacity of V2. Since we repeatedly transfer water from V1 to V2 until V2 becomes full, we will have 'a – b (mod a)' water in V1 when V2 becomes full first time. Once V2 becomes full, it is emptied. We will have 'a – 2b (mod a)' water in V1 when V2 is full second time. We repeat the above steps, and get 'a – nb (mod a)' water in V1 after the vessel V2 is filled and emptied 'n' times. We need to prove that the value of 'a – nb (mod a)' will be 1 for a finite integer 'n'. To prove this, let us consider the following property of coprime numbers.

For any two **coprime integers** 'a' and 'b', the integer 'b' has a **multiplicative inverse** modulo 'a'. In other words, there exists an integer 'y' such that  $b \cdot y \equiv 1 \pmod{a}$  (See 3rd point [here](#)). After '(a – 1)\*y' iterations, we will have 'a – [(a – 1)\*y\*b (mod a)]' water in V1, the value of this expression is 'a – [(a – 1) \* 1] mod a' which is 1. So the algorithm converges and we get 1 litre in V1.