Yen Pham

ID: 11290771

CSCE 2100

Instructor: ABM Rezbaul Islam

Project 1 - Two Water Jugs Problem

Design Description

I. Problem Description

For this project 1, I need determine if and how a certain amount of water can be obtained using two jugs. The user will input an amount of water each jug can hold and an amount of water they require. The program should output if that requirement can be met or not using the available jugs, and the steps necessary to do so.

II. Solving steps

To solve the 2 jugs problem, we would need to know how to determine if the target water amount can be reached given 2 water jugs of any size. Let m be the number of steps to fill or empty Jug A, and n be the number of steps to fill or empty Jug B. The problem can be described in a function as below:

$$JugA * m + JugB * n = \text{water goal} \ (m, n \in Z)$$

$m < 0$: empty jug A

$m > 0$: fill jug A

$n < 0$: empty jug B

$n > 0$: fill jug B

For example, if Jug A's capacity is 7, Jug B's capacity is 3, and water goal is 9, we can fill up Jug B three times while we do not have to fill up jug A to obtain the water goal, as 7 * 0 + 3 * 3 = 9 (when m = 0, n = 3). The steps to obtain the water goal can be described as below:

1. Fill jug B the 1st time: A = 0, B = 3
2. Pour water from jug B to jug A: A = 3, B = 0
3. Fill jug B the 2nd time: A = 3, B = 3
4. Pour water from jug B to jug A: A = 6, B = 0
5. Fill jug B the 3rd time: A = 6, B = 3

Water goal achieved! Total number of times to fill jug B is 3.

For the given water goal and 2 jugs' capacity, solution (m, n) is not unique. There are another way to reach the water goal by filling up jug A 3 times and empty jug B 4 times, as 7 * 3 + 3 * (-4) = 9. The steps to obtain the water goal can be described as below:

1. Fill jug A the 1st time: A = 7, B = 0
2. Pour water from jug A to jug B: A = 4, B = 3
3. Empty jug B the 1st time: A = 4, B = 0
4. Pour water from jug A to jug B: A = 1, B = 3
5. Empty jug B the 2nd time: A = 1, B = 0

6. Pour water from jug A to jug B: A = 0, B = 1
7. Fill jug A the 2nd time: A = 7, B = 1
8. Pour water from jug A to jug B: A = 5, B = 3
9. Empty water from jug B the 3rd time: A = 5, B = 0
10. Pour water from jug A to jug B: A = 2, B = 3
11. Empty jug B the 4th time: A = 2, B = 0
12. Pour water from jug A to jug B: A = 0, B = 2
13. Fill jug A the 3rd time: A = 7, B = 2

Water goal achieved! Total number of times to fill up jug A is 3, total number of times to empty jug B is 4.

From the example, it is clear to see that if there exists integer m, n where JugA * m + JugB * n = water goal, we can fill one jug, pour water from that jug to the other jug, then check with the water goal and keep pouring water until jug A has m times being filled/emptied and jug B has n times being emptied/filled. However, because there are two variable m, n in the function while we only have one function, we cannot solve the function using the traditional algebraic ways. We need to find another way to determine if integer m, n exists for every given jug's capacity and water goal.

There are many strategies to recognize if the water goal can be obtained using 2 given jug sizes. We can check if each filling/emptying/pouring step that we make matches with any of the previous steps. If so, the loops should be terminated and we can conclude that water goal cannot be obtained with the given 2 jug sizes. We can also use greatest common divisor of Jug A and Jug B to decide if water goal can be reached using 2 given water jugs. Let gcd (Jug A, Jug B) be the greatest common divisor of jug A and jug B. By the Bézout's identity, the greatest common divisor of two non- zero number a and b is the smallest positive linear combination of a and b, and an integer is a linear combination of a and b if an only if it is a multiple of their greatest common divisor. As gcd (Jug A, Jug B) is the smallest positive linear combination of Jug A's and Jug B's capacity, to make integer m, n exists (in other words, to make water goal be the linear combination of Jug A and Jug B, or to make function JugA * m + JugB * n = water goal get solutions), water goal must be a multiple of gcd (Jug A, Jug B). Therefore, if water goal % gcd (Jug A, Jug B) = 0, we can determine that the function Jug A * m + Jug B *n = water goal has solution m, n where m, n are integers.

Proof:

∀a € Z (a * gcd (Jug A, Jug B) = Jug A) (by Definition of Greatest Common Divisor)

∀b € Z (b * gcd (Jug A, Jug B) = Jug B) (by Definition of Greatest Common Divisor)

Assume integers m, n exist and water goal is the linear combination of Jug A and Jug B

We have: JugA * m + JugB * n = water goal

↔ a * gcd (Jug A, Jug B) * m + b * gcd (Jug A, Jug B) * n = water goal

↔ gcd (Jug A, Jug B) * (a*m + b * n) = water goal

↔ water goal % gcd (Jug A, Jug B) = 0 (a*m + b*n is a positive integer, as a, m, b, n € Z*)

Because greatest common divisor will help us immediately know if the water goal can be achieved or not, implementing it in this program will help to save some space while running the program.

III. Data Structure

Class Jug
private:
water_capacity (int)
water_value (int)
public:
Initialize class Jug with capacity and initial water value
Get water capacity (int)
Get water value (int)
Fill water until full (void)
Empty water jug (void)
Check if water jug is full (bool)
Check if water jug is empty (bool)
Pour water from the other jug (void)

Other functions:

_Get greatest common divisor of Jug A and Jug B (int)

_Check if the water amount g request can be done with 2 given jugs capacity a and b (bool)

_Present steps of filling/emptying/pouring to reach the water goal (void)

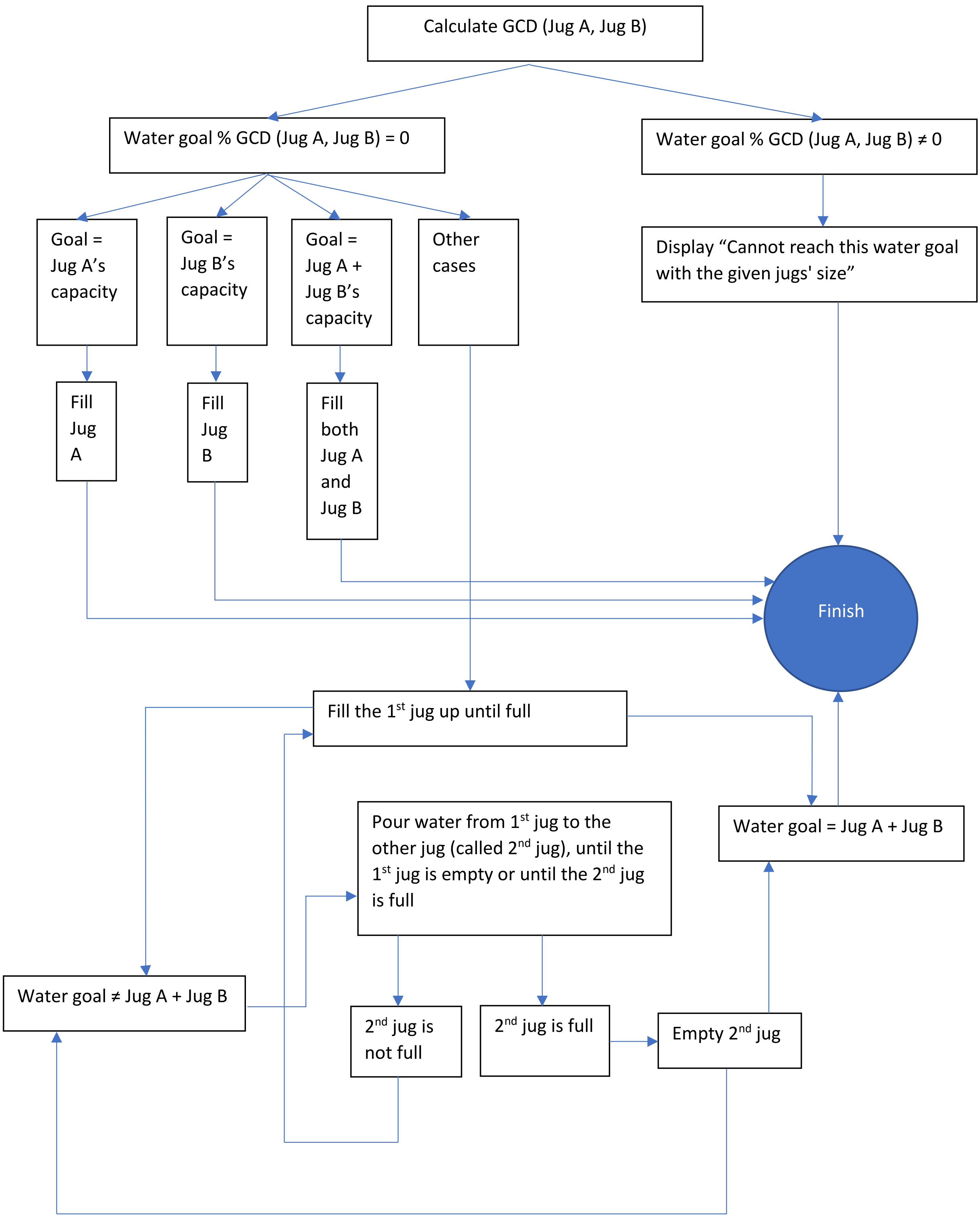
IV. Algorithms

1. Get the greatest common divisor of Jug A's and Jug B's capacity. Go to step 2
2. If water goal % greatest common divisor of Jug A and Jug B = 0, go to step 3. Else, display "cannot reach this goal with the given jugs" and terminate.
3. Once we decided that the water can be reached, here are steps to reach the water goal:
    a. Solve simple cases:
        i. If goal = Jug A's capacity, fill A
        ii. If goal = Jug B's capacity, fill B
        iii. If goal = Jug A+ Jug B, fill both Jugs
    b. If water goal is different from these cases above:
        i. Fill a random Jug until full, we call it 1st jug. Check:
            1. If water goal is reached -> Done
            2. If not, go to 3bii
        ii. Pour water from the 1st jug to the other jug (we call the other jug the 2nd jug), check:
            1. If water goal has not been reached and the 2nd jug has not been full, go back to step 3bi – fill up 1st jug until full then pour from 1st jug to 2nd jug
            2. If water goal has not been reached but 2nd jug is full, empty the 2nd jug. Then, check:
                a. If water goal is reached -> Done
                b. If not, go back to step 3bii - just pour any water amount left from 1st jug to 2nd jug

V. Limitations

Although the program can decide whether water goal can be obtained using 2 given jug sizes and demonstrate the steps to reach the water goal, the program has not shown the minimum steps (where m + n is the smallest) to obtain the water goal. This design is not the best one, however still meets the minimum requirements for this project.

VI. Diagram


Calculate GCD (Jug A, Jug B)
Water goal % GCD (Jug A, Jug B) = 0
Water goal % GCD (Jug A, Jug B) ≠ 0
Goal = Jug A's capacity
Goal = Jug B's capacity
Goal = Jug A + Jug B's capacity
Other cases
Display "Cannot reach this water goal with the given jugs' size"
Fill Jug A
Fill Jug B
Fill both Jug A and Jug B
Finish
Fill the 1st jug up until full
Pour water from 1st jug to the other jug (called 2nd jug), until the 1st jug is empty or until the 2nd jug is full
Water goal = Jug A + Jug B
Water goal ≠ Jug A + Jug B
2nd jug is not full
2nd jug is full
Empty 2nd jug