CheggSolutions - Thegdp

```html

## Discrete Mathematics and Algorithms

### Topic: Integer Pair Transformation

**Given:**
Consider a pair of integers (a, b). The following operations can be performed on (a, b) in any order, zero or more times:

1. (a, b) ➔ (a + b, b)
2. (a, b) ➔ (a, a + b)

The task is to return a string that denotes whether or not (a, b) can be converted to (c, d) by performing the operations zero or more times.

**Example:**

- Input: (a, b) = (1, 1)
- Output: (c, d) = (5, 2)

**Solution Outline:**

1. **Given and Initial Analysis:**
   Given (a, b) and (c, d), it is possible to transform (a, b) into (c, d) if and only if (c, d) can be derived using the allowed operations.
2. **Reverse Transformations:**
   To check if (c, d) can be reached from (a, b), consider reversing the operations:
   - If c > d, consider reversing (c, d) as (c - d, d)
   - If d > c, consider reversing (c, d) as (c, d - c)
3. **Recursion Base Case:**
   The recursion can stop if:
   - If c < a or d < b, it's impossible to transform.
   - If c = a and d = b, the transformation is achieved.
4. **Implementation Strategy:**
   Write a recursive function that checks the possibility of transforming (c, d) back to (a, b) using the reverse operations.

**Step-by-Step Solution:**

1. **Initialization and Input:**

```
def isConvertible(a, b, c, d):
    if c < a or d < b:
        return "No"

    if a == c and b == d:
        return "Yes"

    if c > d:
        return isConvertible(a, b, c - d, d)
    else:
        return isConvertible(a, b, c, d - c)

Define the pairs
```
```

```
a, b = 1, 1
c, d = 5, 2

# Run the check and print the result
print(isConvertible(a, b, c, d))
```

2. **Explanation of the Code:**
   - The code defines the function **isConvertible** that checks if (a, b) can be transformed into (c, d).
   - **Base case 1:** Checks if (c < a or d < b), returning "No" because reverse transformation can't lower values below originals.
   - **Base case 2:** Checks if initial and target pairs are equal, returning check successful "Yes".
   - **Recursive Cases:** Applies reverse transformations until one of the base cases is met.
3. **Detailed Explanation:**
   - **Initial Check:** If either component of (c, d) is smaller than (a, b), print "No" indicating an invalid transformation pathway.
   - **Exact Match:** If both pairs match, transformation possible.
   - **Recursive Reverse Transform:** Depending on whether c > d or vice versa, recursively adjusting c or d, ensuring each step considers both paths to potentially reach the target pair.
4. **Final Solution:** Execute **isConvertible** function with arguments a=1, b=1, c=5, d=2. The expected print result is "Yes".

Conclusion:

This approach ensures that the proposed transformations and logic detect the validity of transforming any given pair of integers (a, b) to another pair (c, d) using allowed operations effectively.

```
```