All possible Ways

Problem Level: Medium

Problem Description:

Given two numbers a and b, the task is to find the number of ways that ***a*** can be represented as the sum of unique integers raised to the power ***b***.

Example 1:

```
Input: a=29 b=2
Output: 2
Explanation: The following are the possible combinations:
```

- $2^2+3^2+4^2 = 29$
- $2^2+5 = 29$

Example 2:

```
Input: a=100 b=3
Output: 1
Explanation: 100 can be represented as cubes of 1,2,3,4.
```

- (1+8+27+64)=100

Approach to be followed:

A good approach to this problem is to use recursive backtracking.

Points to keep in mind:

- Each term has to be strictly greater than the previous term, so we always start checking from the previous number plus one. (At the very beginning, we start at **1**.)
- We backtrack if the sum of the current expression exceeds "**a**", since adding more terms will just increase the sum even further.
- We only go deeper if the current sum so far is strictly smaller than "**a**".
- On the (rare) occasion that we get a sum of exactly "**a**", it means we've found a solution! This also means that we should backtrack (since adding more terms will just increase the sum).

One thing that we have to keep in mind is to make sure not to count the same combinations multiple times. For example: (1,2,3) and (2,3,1) are the same. This is achieved by the first point mentioned above.

Implementation:

```
int getAllWaysHelper(int a, int b, int currNum, int currSum) {
    int result = 0;

    // Calling power of 'i' raised to 'b'
    int p = power(currNum, b);

    while (p + currSum < a) {
        // Recursively check all greater values of 'i'
        result += getAllWaysHelper(a, b, currNum + 1, p + currSum);
        currNum++;
        p = power(currNum, b);
    }

    /*
        If sum of powers is equal to 'a'
        then increase the value of result
    */
    if (p + currSum == a) {
        result++;
    }

    // Return the final result
    return result;
}

int getAllWays(int a, int b) {
    return getAllWaysHelper(a, b, 1, 0);
}
```

Time Complexity: $O(a^{(1/b)})$
Space Complexity: $O(a^{(1/b)})$

Example:
For the example - a=10, b=2.
Let’s do a dry run.

- Try 1 first. Our expression currently looks like $1^2=1$. The next number must be >=2.
    - Try 2. Now it looks like $1^2+2^2=5$. The next number must be greater than >=3.
        - Try 3. Now sum is $1^2+2^2+3^2=14$. **Backtrack, since it exceeds 10.**
    - Try 3. We have $1^2+3^2=10$. **We have found a solution.**
- Try 2. We have $2^2=4$.
    - Try 3. $2^2+3^2=13$. **Backtrack, since it exceeds 10.**
- Try 3. We have $3^2=9$.
    - Try 4. $3^2+4^2=25$. **Backtrack, since it exceeds 10.**
- Try 5. $5^2=25$. **Backtrack, since it exceeds 10.**

After the last backtrack, the enumeration ends.