MCS 253P Quiz 2

Name: ASWIN **UciNetID:** saswin

Write a function "wordPattern", given a pattern and a string s, find if s follows the same pattern.

Here follow means a full match, such that there is a bijection between a letter in pattern and a non-empty word in s.

Examples:

Input: pattern = "abba", s = "dog cat cat dog"
Output: true

Input: pattern = "abba", s = "dog cat cat fish"
Output: false

Input: pattern = "aaaa", s = "dog cat cat dog"
Output: false

Write one edge case: T1: Both match. s = "dog cat bat" p = "a b c"
T2: word & pattern not match s = "cat cat" p = "ab"
T3: ~~Pattern~~ Length not match s = "dog cat" p = abcd

Time Complexity: O(n)
n = pattern length

Your Code: logic: ① Convert string to array of words
② Check if pattern length & number of words are equal
③ Traverse through the words & encode it with pattern with the help of hashmap
④ If for already existing pattern, word doesn't match return false & vice versa
⑤ If both matches return True if we reached end of pattern.

```
vector<string> convertStringToWords(string s)
{   int start = 0, end = 0;
    vector<string> words;
    while (end != s.length()){
        if (s[i] == ' '){
            ans.push_back(
                s.substr(start, end)
            );
            start = end+1
        }
        end++;
    }
    return words;
}
```

```
bool wordPattern(string pattern, string s){
    vector<string> words = convertStringToWords(s);
    int wlen = words.size(), plen = pattern.size();
    if(wlen != plen) return false;
    unordered_map<string, char> wtp;
    unordered_map<char, string> ptw;
    for(int i=0; i<plen; i++){
        string word = words[i], char p = pattern[i];
        //check if already exist.
        if(wtp.find(word) != wtp.end())
            if(wtp[word] != pattern[i]) return false;
        if(ptw.find(p) != ptw.end())
            if(ptw[p] != word) return false;
        ptw[p] = word; wtp[word] = p;
    }
    return true;
}
```