All palindromic partition in C++

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

class AllPalindromicPartition {
public:
    static void main() {
        string str = "abba";
        sol(str, "");
    }

    static void sol(string str, string asf) {
        if (str.length() == 0) {
            cout << asf << endl;
            return;
        }

        for (int i = 0; i < str.length(); i++) {
            string prefix = str.substr(0, i + 1);
            string ros = str.substr(i + 1);
            if (isPalin(prefix)) {
                sol(ros, asf + "(" + prefix + ")");
            }
        }
    }

    static bool isPalin(string s) {
        int li = 0;
        int ri = s.length() - 1;
        while (li < ri) {
            if (s[li] != s[ri]) {
                return false;
            }
            li++;
            ri--;
        }
        return true;
    }
};

int main() {
    AllPalindromicPartition::main();
    return 0;
}
```

Dry Run for Input: "abba"

1st Level of Recursion:

- prefix = "a", ros = "bba", and "a" is a palindrome.
- Call sol("bba", "(a)").

2nd Level of Recursion:

- prefix = "b", ros = "ba", and "b" is a palindrome.
- Call sol("ba", "(a)(b)").

3rd Level of Recursion:

- prefix = "b", ros = "a", and "b" is a palindrome.
- Call sol("a", "(a)(b)(b)").

4th Level of Recursion:

- prefix = "a", ros = "", and "a" is a palindrome.
- Output (a)(b)(b)(a).

Backtracking to Explore Other Partitions:

3rd Level (Exploring Longer Prefixes):

- prefix = "bb", ros = "a", and "bb" is a palindrome.
- Call sol("a", "(a)(bb)").

4th Level:

- prefix = "a", ros = "", and "a" is a palindrome.
- Output (a)(bb)(a).

Backtracking to 2nd Level:

- prefix = "bba" is not a palindrome. Skip.

Backtracking to 1st Level:

- prefix = "ab" and prefix = "abb" are not palindromes.
- prefix = "abba", ros = "", and "abba" is a palindrome.

	• Output (abba).
Output:- (a)(b)(b)(a) (a)(bb)(a) (abba)	