ZigZag Conversion

The string "PAYPALISHIRING" is written in a zigzag pattern on a given number of rows like this: (you may want to display this pattern in a fixed font for better legibility)

```
P   A   H   N
A P L S I I G
Y   I   R
```

And then read line by line: "PAHNAPLSIIGYIR"

Write the code that will take a string and make this conversion given a number of rows:

string convert(string s, int numRows);
Example 1:

```
Input: s = "PAYPALISHIRING", numRows = 3
Output: "PAHNAPLSIIGYIR"
```

Example 2:

```
Input: s = "PAYPALISHIRING", numRows = 4
Output: "PINALSIGYAHRPI"
```

Explanation:

```
P   I   N
A L S I G
Y A H R
P   I
```

Solution 1 in C++

```
#include <iostream>


using namespace std;

class Solution {
public:
    string convert(string s, int numRows) {
        string a[numRows];
        int str_len=s.length();
        int i;
        int pos=0;
        int direction=1;
        int turning_point;
        string ret;

        turning_point=numRows-1;

        for( i=0; i<str_len; i++) {
            a[pos] += s[i];
            if (pos==turning_point) {
                if (numRows<3)
                    pos=0;
                else {
                    direction *= -1;
                    if (direction<0) {
                        turning_point=(numRows>2)?1:0;
                        pos=numRows-2;
                    } else {
                        turning_point=numRows-1;
                        pos=0;
                    }
                }
            }
            else pos += direction;
            cout << s[i] << " " << pos << " " << direction << " " << turning_point << endl;
        }
        for( i=0; i<numRows; i++) {
            ret += a[i];
            cout << a[i] << endl;
        }
        return ret;
    }
};

int main(void)
{
    Solution s;
    string testcase = "PAYPALISHIRING";

    cout << testcase << "(3) -> " << s.convert( testcase, 3) << " : Expected = PAHNAPLSIIGYIR" << endl;

    cout << testcase << "(4) -> " << s.convert( testcase, 4) << endl;

    return 0;
}
```

Solution 2 in C++

```
class Solution {
public:
    string convert(string s, int numRows) {

        if (numRows == 1) return s;

        vector<string> rows(min(numRows, int(s.size())));
        int curRow = 0;
        bool goingDown = false;

        for (char c : s) {
            rows[curRow] += c;
            if (curRow == 0 || curRow == numRows - 1) goingDown = !goingDown;
            curRow += goingDown ? 1 : -1;
        }

        string ret;
        for (string row : rows) ret += row;
        return ret;
    }
};
```

Solution 3 in Java

```
class Solution {
    public String convert(String s, int numRows) {

        if (numRows == 1) return s;

        List<StringBuilder> rows = new ArrayList<>();
        for (int i = 0; i < Math.min(numRows, s.length()); i++)
            rows.add(new StringBuilder());

        int curRow = 0;
        boolean goingDown = false;

        for (char c : s.toCharArray()) {
            rows.get(curRow).append(c);
            if (curRow == 0 || curRow == numRows - 1) goingDown = !goingDown;
            curRow += goingDown ? 1 : -1;
        }

        StringBuilder ret = new StringBuilder();
        for (StringBuilder row : rows) ret.append(row);
        return ret.toString();
    }
}
```