```
/*
给定一个单词列表，只返回可以使用在键盘同一行的字母打印出来的单词。键盘如下图所示。
American keyboard

示例：
输入: ["Hello", "Alaska", "Dad", "Peace"]
输出: ["Alaska", "Dad"]

注意：

    你可以重复使用键盘上同一字符。
    你可以假设输入的字符串将只包含字母。

来源：力扣（LeetCode）
链接：https://leetcode-cn.com/problems/keyboard-row
著作权归领扣网络所有。商业转载请联系官方授权，非商业转载请注明出处。
*/
```

分析:

- 常规方法
  - 建立三个map存放三行字母的映射值;
  - 遍历每组字符串统计行属性情况;
  - 根据行属性统计情况决定是否要把原每组字符压入返回值.

方法一:C++_Map

```
class Solution
{
    private:
        map<char,int> mci1;
        map<char,int> mci2;
        map<char,int> mci3;

        bool isSameLine(string& s)
        {
            int     temp    =   0       ;
            int     i       =   0       ;
            bool    ret_val =   true    ;

            for(i = 0; i < s.size(); i ++)
            {
                if(mci1.count(s.at(i)))
                {
                    temp |= 0x01;
                }
                else if(mci2.count(s.at(i)))
                {
                    temp |= 0x02;
                }
                else
```

```
                {
                    temp |= 0x04;
                }
            }

            switch(temp)
            {
                case 0x01:
                case 0x02:
                case 0x04:
                    return true;
                break;
                default:
                    return false;
                break;

            }
        }


    public:
        vector<string> findWords(vector<string>& words)
        {
            vector<string>  ret_val             ;
            int             i           =   0   ;

            for(i=0;i<words.size();i++)
            {
                if(isSameLine(words[i]))
                {
                    ret_val.push_back(words[i]);
                }
            }
            return ret_val;
        }

    Solution()
    {
        /* qwertyuiop */
        mci1['q']   =   0;
        mci1['w']   =   1;
        mci1['e']   =   2;
        mci1['r']   =   3;
        mci1['t']   =   4;
        mci1['y']   =   5;
        mci1['u']   =   6;
        mci1['i']   =   7;
        mci1['o']   =   8;
        mci1['p']   =   9;

        mci1['Q']   =   10;
        mci1['W']   =   11;
        mci1['E']   =   12;
        mci1['R']   =   13;
        mci1['T']   =   14;
        mci1['Y']   =   15;
        mci1['U']   =   16;
        mci1['I']   =   17;
```

```
        mci1['O']   =   18;
        mci1['P']   =   19;

        /* asdfghjkl */
        mci2['a']   =   20;
        mci2['s']   =   21;
        mci2['d']   =   22;
        mci2['f']   =   23;
        mci2['g']   =   24;
        mci2['h']   =   25;
        mci2['j']   =   26;
        mci2['k']   =   27;
        mci2['l']   =   28;
        mci2['A']   =   30;
        mci2['S']   =   31;
        mci2['D']   =   32;
        mci2['F']   =   33;
        mci2['G']   =   34;
        mci2['H']   =   35;
        mci2['J']   =   36;
        mci2['K']   =   37;
        mci2['L']   =   38;

        /* zxcvbnm */
        mci3['z']   =   40;
        mci3['x']   =   41;
        mci3['c']   =   42;
        mci3['v']   =   43;
        mci3['b']   =   44;
        mci3['n']   =   45;
        mci3['m']   =   46;

        mci3['Z']   =   50;
        mci3['X']   =   51;
        mci3['C']   =   52;
        mci3['V']   =   53;
        mci3['B']   =   54;
        mci3['N']   =   55;
        mci3['M']   =   56;
    }
};

/*
执行结果:
通过
显示详情
执行用时 :4 ms, 在所有 C++ 提交中击败了74.02% 的用户
内存消耗 :8.8 MB, 在所有 C++ 提交中击败了5.80%的用户
*/
```

AlimyBreak
2019.09.09