# 题目

给定一组单词words，编写一个程序，找出其中的最长单词，且该单词由这组单词中的其他单词组合而成。若有多个长度相同的结果，返回其中字典序最小的一项，若没有符合要求的单词则返回空字符串。

示例：

输入： ["cat","banana","dog","nana","walk","walker","dogwalker"]

输出： "dogwalker"

解释： "dogwalker"可由"dog"和"walker"组成。

提示：

0 <= len(words) <= 200

1 <= len(words[i]) <= 100

# 分析

## 方法一：哈希表

思路：

使用哈希集合存储列表中的每一个单词，接着遍历列表的中的每个单词判断其是否由其他单词组合而成

递归查找时，因为不能由其自身组成，因此每次从哈希集合中删除其自身

递归分割查找每一种可能，当一种分割到空串时，返回true，所有的情况遍历完毕，返回false

代码：

class Solution {

public:

string longestWord(vector<string>& words) {

unordered\_set<string> allWords(words.begin(), words.end());

string ans;

for(auto word:words)

{

auto tmpCollects = allWords;

tmpCollects.erase(word);

if(isCombinated(word,tmpCollects))

{

if (word.size() > ans.size())ans = word;

if (word.size() == ans.size())ans = min(ans, word);

}

}

return ans;

}

private:

bool isCombinated(string s,unordered\_set<string>& words)

{

if (s.size() == 0)return true;

for(int i=1;i<=s.size();i++)

{

if (words.count(s.substr(0, i)) && isCombinated(s.substr(i), words))return true;

}

return false;

}

};