class Solution {

public List<String> letterCombinations(String digits) {

List<String> result=new ArrayList<String>();

if(digits==null||digits.length()==0) return result;

//设置边界，error情况，绝大部分function第一步都是这个

String[] mapping={

"0","1","abc","def","ghi","jkl","mno","pqrs","tuv","wxyz"

};

//在recursive之前，我们把需要的不变的参数准备好

letterCombinationsRecursive(result,digits,mapping,"",0);

//开始call recursive参数

return result;

}

public void letterCombinationsRecursive(List<String> result,String digits, String[] mapping, String current, int index){

if (index==digits.length()){

result.add(current);

return;

} //base case

String letters=mapping[digits.charAt(index)-'0'];

for(int i=0;i<letters.length();i++)

letterCombinationsRecursive(result,digits,mapping,current+letters.charAt(i),index+1);

}

}

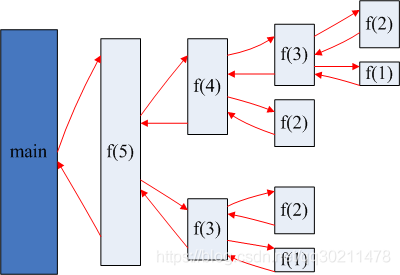
什么时候使用recursion

一个功能在被重复使用，并每次使用时，参与运算的结果与上一次调用有关，此时可以用递归来解决问题。

1.有边界

2.递归前进段：当前的参数不等于1的时候，继续调用自身，也就是继续call的箭头

3.递归返回段，也就是返回箭头，我们碰到终点开始收获了



我们可以画图辅助理解，如果

recursion题目标准格式：

1.因为标准参数是限死的，没法更替参数达到recursion的效果，因此我们只把他当做driver,



然后我们在下面function之外，class之内，写recursive function

Recursion内部

2.先写base case: 代表着到底了，