第六周学习总结

1．单词搜索 2 用 Tire 树方式实现的时间复杂度：O(m*n*4^k)

2. 双向 BFS 模板

```
while front and back:
    k += 1
    next_front = set()
    for word in front:
        for i in range(word_len):
            for c in string.lowercase:
                new_word = word[ : i ] + c + word[ i + 1 : ]
                if new_word in back:
                    return k
                if new_word in wordlist:
                    new_front.add(new_word)
                    wordlist.remove(new_word)
    front = new_front
    if len(back) < len(front):
        front, back = back, front                    双向 bfs 判断条件
return 0
```

set 类型 next_front 为缓存，每个循环（即每层 BFS）清空 next_front 从 front 或 back 开始遍历