实现copy函数

指定一个源文件，实现copy到目标目录。

例如把/tmp/test.txt 拷贝到 /tmp/test1.txt

```
filename1 = '/tmp/test.txt'
filename2 = '/tmp/test1.txt'

f = open(filename1, 'w+')
lines = ['abc', '123', 'magedu']
f.writelines('\n'.join(lines))
f.seek(0)
print(f.read())
f.close()


def copy(src, dest):
    with open(src) as f1:
        with open(dest, 'w') as f2:
            f2.write(f1.read())

copy(filename1, filename2)
```

单词统计

有一个文件，对其进行单词统计，不区分大小写，并显示单词重复最多的10个单词。

```
filename = 'sample.txt'
d = {}
with open(filename, encoding='utf8') as f:
    for line in f:
        words = line.split()
        for word in map(str.lower, words):
            d[word] = d.get(word, 0) + 1

print(sorted(d.items(), key=lambda item: item[1], reverse=True))

# 或使用缺省字典
from collections import defaultdict

d = defaultdict(lambda :0)
with open(filename, encoding='utf-8') as f:
    for line in f:
        words = line.split()
        for word in map(str.lower, words):
            d[word] += 1

print(sorted(d.items(), key=lambda item: item[1], reverse=True))
```

```
# 得数如下：
# the, 136
# is, 60
# a, 54
# path, 52
# if, 42
# and, 39
# to, 34
# of, 33
# on, 32
# return, 30
```

这是帮助文档中path的文档，path应该很多。

```
for k in d.keys(): # 从key里面看看还有好多带有path的
    if k.find('path') > -1:
        print(k)
```

使用上面的代码，就可以看到path非常多
os.path.exists(path) 可以认为含有2个path。

```
# 思路：遇到特殊字符，就用空格替代
def makekey(s:str):
    chars = set(r"""!'"#./\()[],*-""")
    key = s.lower()
    ret = []
    for i, c in enumerate(key):
        if c in chars:
            ret.append(' ')
        else:
            ret.append(c)
    return ''.join(ret).split()

d = {}
with open('sample', encoding='utf8') as f:
    for line in f:
        words = line.split()
        for wordlist in map(makekey, words):
            for word in wordlist:
                d[word] = d.get(word, 0) + 1

for k,v in sorted(d.items(), key=lambda item: item[1], reverse=True):
    print(k,v)

# 对单词做进一步处理后，统计如下：
path 138
the 136
is 60
a 59
os 49
```

```
if 43
and 40
to 34
on 33
of 33
```

分割key的另一种思路

```
def makekey(s:str):
    chars = set(r"""!'"#./\()[],*-""")
    key = s.lower()
    ret = []
    start = 0
    length = len(key)

    for i, c in enumerate(key):
        if c in chars:
            if start == i: # 如果紧挨着还是特殊字符，start一定等于i
                start += 1 # 加1并continue
                continue
            ret.append(key[start:i])
            start = i + 1 # 加1是跳过这个不需要的特殊字符c
    else:
        if start < len(key): # 小于，说明还有有效的字符，而且一直到末尾
            ret.append(key[start:])

    return ret

print(makekey('os.path.exists(path)'))
print(makekey('os.path.-exists(path))'))
print(makekey('path.os...'))
print(makekey('path'))
print(makekey('path-p'))
print(makekey('***...'))
print(makekey(''))
```

完整代码

```
def makekey(s:str):
    chars = set(r"""!'"#./\()[],*-""")
    key = s.lower()
    ret = []
    for i, c in enumerate(key):
        if c in chars:
            ret.append(' ')
        else:
            ret.append(c)
    return ''.join(ret).split()

def makekey1(s:str):
    chars = set(r"""!'"#./\()[],*-""")
    key = s.lower()
    ret = []
    start = 0

    for i, c in enumerate(key):
        if c in chars:
            if start == i: # 如果紧挨着还是特殊字符，start一定等于i
                start += 1 # 加1并continue
                continue
            ret.append(key[start:i])
            start = i + 1 # 加1是跳过这个不需要的特殊字符c
    else:
        if start < len(key): # 小于，说明还有有效的字符，而且一直到末尾
            ret.append(key[start:])

    return ret


d = {}
with open('sample', encoding='utf8') as f:
    for line in f:
        words = line.split()
        for wordlist in map(makekey1, words):
            for word in wordlist:
                d[word] = d.get(word, 0) + 1

for k,v in sorted(d.items(), key=lambda item: item[1], reverse=True):
    print(k,v)
```

以上代码还可以再优化，重新封装。