```python
# 1.写出完整的装饰器(不用考虑带参装饰器，就是普通装饰器)语法
def wrapper(func):
    def inner(*args, **kwargs):
        pass
        result = func(*args, **kwargs)
        pass
        return result
    return inner
```

```python
# 2.有一个计算两个数和的方法，为其添加一个确保两个参数都是int或float类型的装饰器，保证运算不会抛异常
def check_num(func):
    def inner(n1, n2):
        if not (isinstance(n1, (int, float)) and isinstance(n2, (int, float))):
            print('不能求和')
            return  # 结束掉，不让其进入计算功能
        return func(n1, n2)
    return inner

@check_num
def add(n1, n2):
    return n1 + n2

print(add(1, '2'))
```

```python
# 3.有一个一次性录入人名并返回人名的方法(人名只考虑存英文)，为其添加一个装饰器，使得处理后人名首字母一定大写
def upper_name(func):
    def inner():
        result = func()
        return result.title()  # Owen
    return inner

@upper_name
def get_name():
    name = input('name: ')  # owen
    return name

print(get_name())
```

拓展题：

1.原功能：entry_grade
*）可以完成『成绩录入功能』
    -- 可以重复录入成绩，默认所有输入都是合法的(1~100之间的数)
    -- 当录入成绩为0时，结束成绩的录入
    -- 将录入的成绩保存在列表中并返回给外界，eg：[90, 80, 50, 70]

2.选择课程装饰器：choose_course
*）为『成绩录入功能』新增选择课程的拓展功能，达到可以录入不同学科的成绩
    -- 可以重复输入要录制的学科名，然后就可以进入该门学科的『成绩录入功能』，录入结束后，可以进入下一门学科成绩录入
    -- 当输入学科名为q时，结束所有录入工作
    -- 将学科成绩保存在字典中并返回给外界，eg：{'math':[90, 80, 50, 70], 'english':[70, 50, 55, 90]}

3.处理成绩装饰器：deal_fail
*）可以将所有录入的成绩按60分为分水岭，转换为 "通过" | "不通过" 进行存储
    -- 如果只对原功能装饰，结果还为list返回给外界，eg：["通过", "通过", "不通过", "通过"]
    -- 如果对已被选择课程装饰器装饰了的原功能再装饰，结果就为dict返回给外界，eg：{'math':["通过", "通过", "不通过", "通过"], 'english':["通过", "不通过", "不通过", "通过"]}

```python
# 选择课程装饰器
def choose_course(func):
    def inner(*args, **kwargs):
        grade_map = {}  # {'math':[math的成绩们], 'english':[english的成绩们]}
        while True:
            course = input('course: ')  # 循环录入学科
            if course == 'q':  # q代表退出录入
                break
            result = func(*args, **kwargs)  # 进入录入成绩功能可以完成该学科成绩录入
            grade_map[course] = result  # 形成 学科：[成绩们] 的k-v键值对
        return grade_map
    return inner


# 处理成绩装饰器
def deal_fail(func):
    def inner(*args, **kwargs):
        result = func(*args, **kwargs)  # 先获得成就结果
        if isinstance(result, list):  # 没有被录入学科装饰的分支
            for i, v in enumerate(result):  # enumerate可以为可迭代对象添加迭代索引
                if v < 60:
                    result[i] = '不通过'
                else:
                    result[i] = '通过'
        if isinstance(result, dict):  # 被录入学科装饰的分支
```

```
            for ls in result.values():
                for i, v in enumerate(ls):
                    if v < 60:
                        ls[i] = '不通过'
                    else:
                        ls[i] = '通过'
        return result
    return inner


@choose_course
@deal_fail
# 成绩录入功能
def entry_grade():
    grades = []
    while True:  # 循环录入成绩
        grade = int(input('grade: '))
        if grade == 0:  # 录入0退出录入
            break
        grades.append(grade)  # 成绩存放到list中
    return grades

grades = entry_grade()
print(grades)
```