MIT 코스웨어
C7 | 디버깅과 예외, Assertion

[대참사를 막으려면]

프로그램에서는 필연적으로 버그가 발생한다.
이 버그를 방지하려면,
1) Defensive Programming
- 모듈화, decom, 주석, assert

2) Testing and Validation
- input/output compare하기, breaking program

3) Debugging
- 에러를 일으키는 이벤트를 분석해서 해결

[디버깅에 대해서]

디버깅을 쉽게 하고 효율적으로 하려면,
1) 애초에 프로그램을 start할때 쉽게 하라
2) 프로그램을 모듈화해서 개별 디버깅이 가능하도록
3) 모듈마다 주석을 달아서 input, output을 예측하기!
4) code에 대해서 assumption을 정하기

차례로 유닛, regression, integration으로 체크하기
테스트의 두 가지 종류는 블랙박스, 글래스박스:
블랙박스 > 그냥 기능상으로 테스트를 진행하기 (코드 리뷰 X)
글래스박스 > 코드 하나씩 테스트 진행하기

글래스박스 상에서는 미싱 패스 문제점이 존재하므로,
각 branch, loop, while loop별로 가이드라인이 존재!
branches > 모든 condition 파악하기
for loops > 루프 진입전, 한번만 진행, 여러번 진행
while loops > 탈출조건까지 루프를 돌려보기

[메시지 에러의 예시]

IndexError: 형성된 인덱스 이상을 벗어나는 것
TypeError: 올바르지 않은 문자열 타입 변환
NameError: 변수 설정하지 않은 경우에 호출한 경우
TypeError: 데이타 타입 간의 올바르지 않은 연산 ('3'/4)
SyntaxError: 문법상의 오류 (괄호 안닫는 경우)

[예외와 가정 도입하기]

파이썬은 try.. except를 통해서 예외를 처리할 수 있다!
try에서 메인 코드를 실행해서 예외 발생시 except를 실행한다.
ValueError, ZeroDivisionError 등의 종류가 있다.
만약 예외가 발생하지 않으면 else문을 실행한다.
finally는 어떤 경우에도 실행하는 조건문이다.
raise을 통해서 예외를 일부러 발생시키면 바로 except로!
오류가 발생할 때, except에서 return으로 기본값을 형성 가능!

프로그램적 가정은 '방어적 프로그래밍' 에 필수적이다.
assert "조건" "메시지" 형태로 디버깅을 실행하는 형식이다.
만약 assertion에 문제가 생기면 AssertionError