/* elice */

도전! 디버깅 입문

에러를 만나도 당황하지 않는 법

김건우 선생님

커리큘럼

3 **나의 첫 테스트 코드**

코드가 바뀌어도 올바르게 동작할 수 있도록 도와 주는
테스트 코드를 작성해 봅니다.

4 **실전 디버깅!**

실전 문제를 풀어 보며,
다양한 버그를 찾아내고 안전한 코드를 설계하는 능력을 기릅니다.

목차

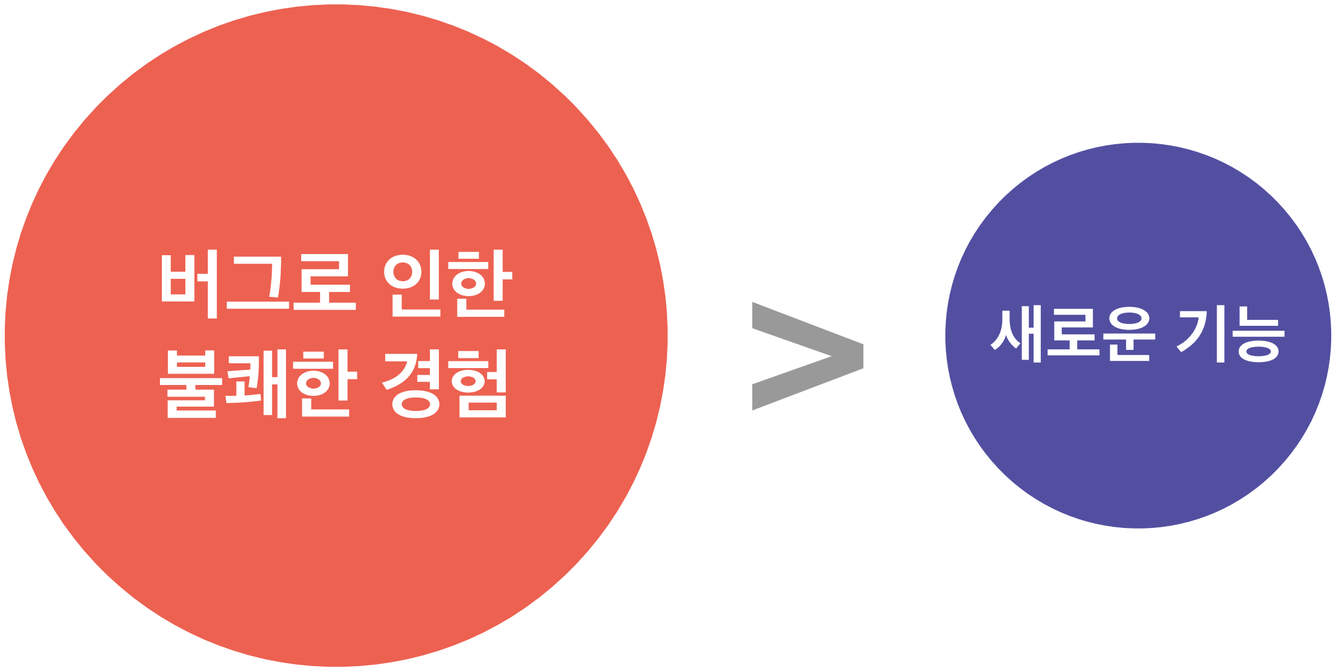
빠른 복습

디버깅이란?

사용/테스트 중 찾아낸 버그를 없애 나가는 과정

- 버그의 **원인**을 찾고
- 발생한 버그를 **해결**하고
- 비슷한 버그의 **재발을 방지**하는 것

사용자 경험과 직결



처음부터 완벽한 코드는 없다

코드가 잘 작동하지 않을 때	어... 왜 안 되지?
코드가 한 번에 작동할 때	어... 왜 벌써 되지?

빠르게 문제를 **파악/해결**하는 것이 능력!

실패는 성공의 어머니

많은 버그를 경험할 수록,

더 **안전한 코드**를 설계하는 능력이 생긴다!

에러 메시지 읽기

```
Traceback (most recent call last):
  File "main.py", line 4, in <module>
    greeting("Donald Trump")
  File "main.py", line 2, in greeting
    print("Hello", yourname + "!")
NameError: name 'yourname' is not defined
```

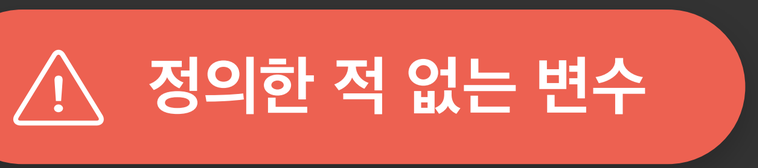
Syntax error

```
def add_all(numbers):
    result = 0
    for number in numbers
        result += number
    return result
```

⚠ 컴퓨터가 이해할 수 없는 코드

Name error

```
def add_all(numbers):
    result = 0
    for number in numbers:
        result += numbre
    return result
add_all([1, 2, 3])
```



Type error

```
def usd_to_krw(price):
    price_in_krw = price * 1100
    return price_in_krw + " won"
usd_to_krw(4.99)
```

⚠ 숫자와 'won'을 더할 수 없음

Index error

```
def first_character(string):
    return string[0]


first_character("")
```

⚠ 문자열의 0번째 글자가 존재하지 않음

Zero division error

```
def average(numbers):
    return sum(numbers) / len(numbers)


average([])
```

⚠ 0으로 나눌 수 없음

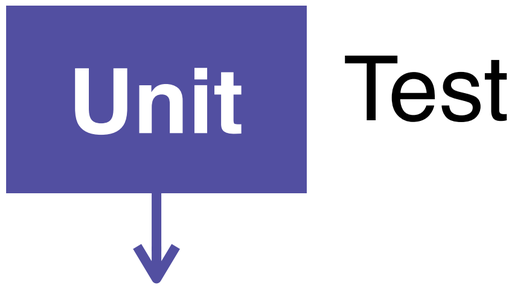
try / except

```
def average(numbers):
    return sum(numbers) / len(numbers)
```

try / except

```
def average(numbers):
    try:
        return sum(numbers) / len(numbers)
    except ZeroDivisionError:
        print("No numbers!")
```

유닛 테스트



가장 작은 단위

(= 함수 1개)

유닛 테스트

테스트 작성을 어떻게 작성하느냐에 따라
어떤 출력을 의도하는지가 결정됨

즉, **테스트 == 설계**

유닛 테스트의 조건

1. 읽기 쉽다
2. 독립적이다
3. 충분히 작다
4. 충분히 넓다

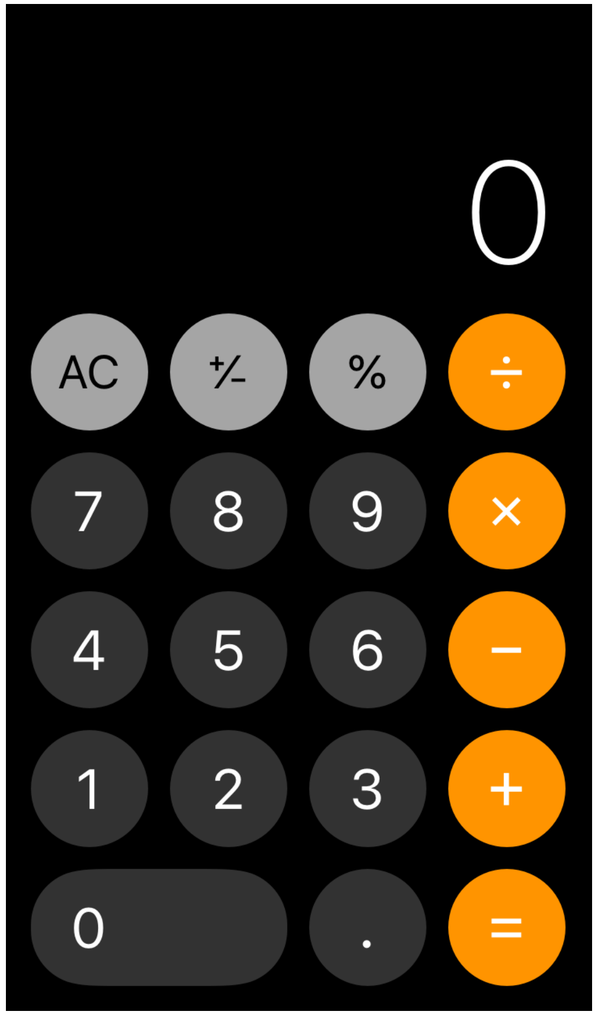
unittest

```
class IsPalindromeTests(unittest.TestCase):
    def test_level(self):
        self.assertTrue(is_palindrome("level"))
    def test_lever(self):
        self.assertFalse(is_palindrome("lever"))


unittest.main()
```

이번 주 프로젝트 소개

계산기



계산기

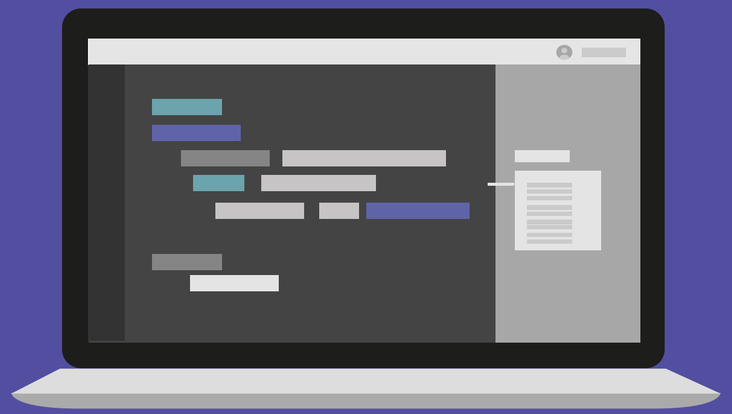
```
class Calculator():
    def clear():
        ...
    def press_digit(digit):
        ...
    def press_plus():
        ...
```

계산기

```
class CalculatorTests(unittest.TestCase):
    def test_press_one_digit(self):
        ...
    def test_press_multiple_digits(self):
        ...
    def test_press_clear_after_writing(self):
        ...
```

계산기 테스트 / 디버깅

[프로젝트] 계산기 디버깅 / 테스트



/* elice */

문의 및 연락처

academy.elice.io

contact@elice.io

facebook.com/elice.io

medium.com/elice