Why are computers inaccurate?

컴퓨터는 왜 부정확할까?

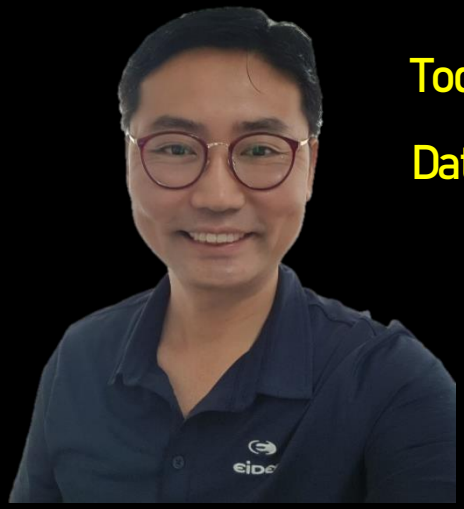
소프트웨어 꼰대 강의

노기섭 교수

(kafa46@gmail.com)

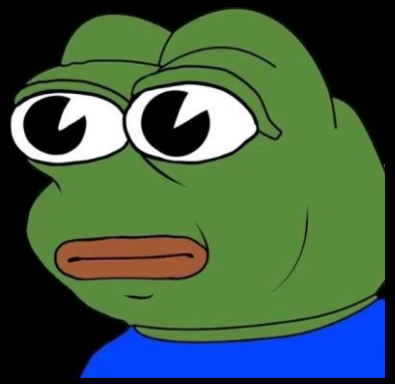
Motivations

In Python programming class



Today's topic is DataType!

DataType is very important.



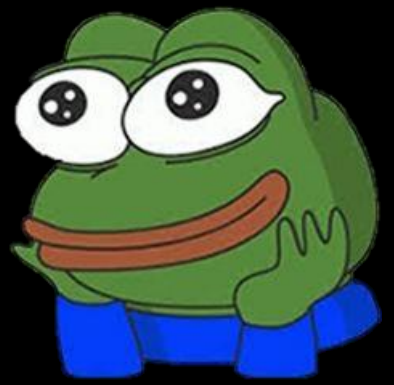
Oh....

Really?

In float datatype,

There always exists some error.

Also, all other programming languages have same problem.



Oh...

Cool...

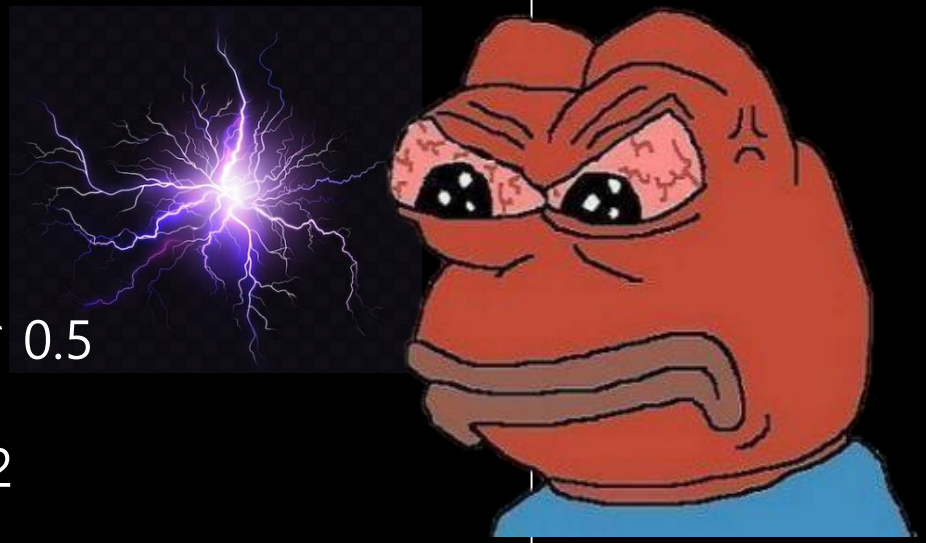
This isn't an error, it's a natural phenomenon that occurs in computers.

Motivations

But! Hey, professor.

It is still correct after float operation!

```
>>> 0.1
0.1
>>> 0.2
0.2
>>> 1/10
0.1
>>> 0.3 * 0.5
0.15
>>> 1/0.2
5.0
>>> 0.5 / 0.4
1.25
```



Let's check these cases

Hey, calm down ^^

How to explain this phenomenon?

Let's check it out more!

```
>>> 0.1
0.1
```

```
>>> 0.1
0.1
```

```
>>> 0.2
0.2
```

```
>>> 0.2
0.2
```

So far, nothing special

```
>>> 1 / 10
0.1
```

```
>>> 1 / 11
0.09090909090909091
```

```
>>> 0.3 * 0.5
0.15
```

```
>>> 0.1 * 0.1
0.010000000000000002
```

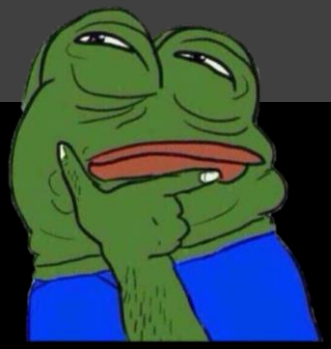
```
>>> 1 / 0.2
5.0
```

```
>>> 1 / 11
0.090909090909090
```

```
>>> 0.5 / 0.4
1.25
```

```
>>> 0.5 / 0.3
1.6666666666666667
```

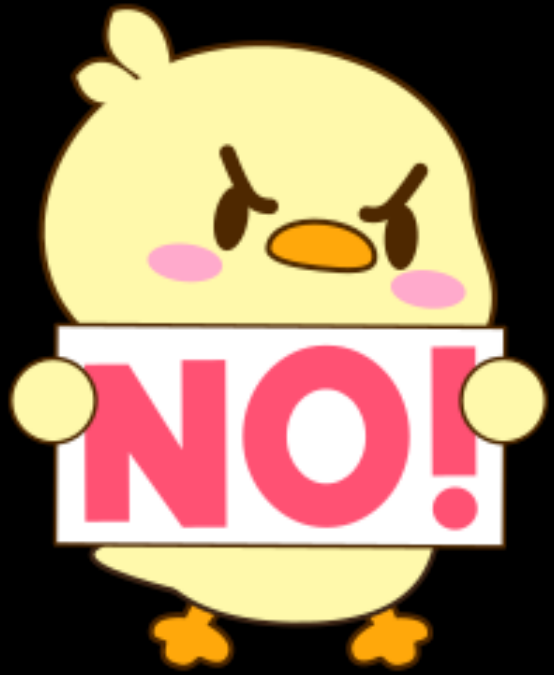
So what?



Hey, prof!

It's just very small error.

I think we may have no problem in computations ^^



- 수많은 거래를 처리하는 금융/세금 시스템
- 정밀한 계산이 필요한 항공 우주 시스템
- 안전 및 정확도가 중요한 방산 시스템

⋮

작은 오차라도 누적되면 치명적 결과

(Toy Example) Tax Computing using Float

```
from decimal import Decimal


price = 10.0    # 가격
discount = 0.1  # 할인율
tax_rate = 0.05 # 세율

# 부동 소수점(float) 연산
price_float_1 = price * (1 - discount) * (1 + tax_rate)
price_float_2 = (price - (price * discount)) + ((price - (price * discount)) * tax_rate)

print(f"최종 가격 (float 계산 1): {price_float_1}")
print(f"최종 가격 (float 계산 2): {price_float_2}")
print(f"최종 가격 동일 여부: {price_float_1 == price_float_2}")
```

```
최종 가격 (float 계산 1): 9.450000000000001
최종 가격 (float 계산 2): 9.45
최종 가격 동일 여부: False
```

Finally, I decided to prepare some lecture

We need to understand the fundamental reason.

From the basic

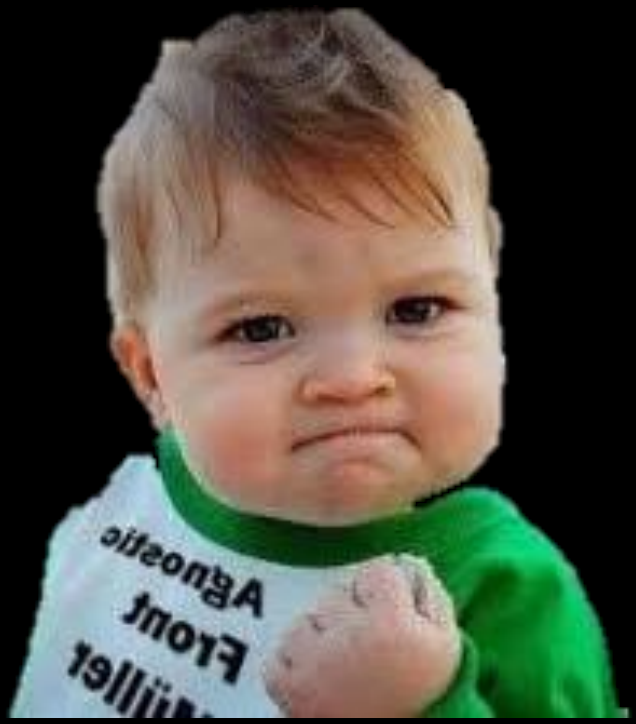
Until complete understanding

Course Overview

Topic	Contents
01. Orientation 오리엔테이션	Course introduction, motivations, final objectives 과정 소개, 동기부여, 최종 목표
02. Converting floating point 실수 변환	How to convert float from decimal to binary 어떻게 십진수를 이진수로 변환하는가?
03. Fixed-point Representation 고정 소수점 방식	How to represent float in fixed representation 어떻게 고정 소수점 방식으로 실수를 표현하는가?
04. Floating-point Representation 부동 소수점 방식	How to represent float in floating representation 어떻게 부동 소수점 방식으로 실수를 표현하는가?
05. Handling Negative Numbers 음수 처리	Complement, Radix, n-ary System, etc. 보수, 기수, 진법 등

References

- **Python Official Docs**
  - Floating-Point Arithmetic: Issues and Limitations
    - URL: https://docs.python.org/3.13/tutorial/floatingpoint.html
- **YouTube**
  - **얄팍한 코딩사전, 부동소수점 (+ 실수계산 오차가 생기는 이유)**
    - https://www.youtube.com/watch?v=ZQDsWySjY6g&t=333s
- **Useful Blogs**
  - 실수 표현(부동 소수점) 원리 한눈에 이해하기
    - URL: https://inpa.tistory.com/entry/JAVA-%E2%98%95-%EC%8B%A4%EC%88%98-%ED%91%9C%ED%98%84%EB%B6%80%EB%8F%99-%EC%86%8C%EC%88%98%EC%A0%90-%EC%9B%90%EB%A6%AC-%ED%95%9C%EB%88%88%EC%97%90-%EC%9D%B4%ED%95%B4%ED%95%98%EA%B8%B0
  - [파이썬] 부동소수점 오차 해결하는 방법(Decimal 라이브러리)
    - https://jeonginyun.tistory.com/109



수고하셨습니다 ..^^..