[item 60] 정확한 답이 필요하다면 float 와 double 은 피하라

Content

> flouat, double

- 과학, 공학 계산용으로 설계되었음
- 이진 부동 소수점 연산에 쓰인다
- 넓은 범위의 수를 빠르게 정밀한 `근사치`로 계산하도록 세심하게 설계 되었음

> 따라서 float 와 double 을 정확한 결과가 필요할 때 사용하면 안된다

- 금융 관련 계산과 맞지 않는다
- 0.1, 0.01, 0.001, ... 등을 표현할 수 없기 때문이다

> 정확한 계산에서는 BigDecimal, int, long 을 사용해야 한다

BigDecimal

- 소수점 추적은 시스템에 맡기고, 코딩시의 불편함이나 성능 저하를 신경쓰지 않아도 되는 경우
- 장점? 정확한 소수점 계산을 할 수 있다.
- 단점? 기본 타입보다 쓰기 불편하고 느리다
- 그 외에, BigDecimal 은 8가지 반올림 모드를 제공한다.

int, long

- 성능이 중요하고 소수점을 직접 추적할 수 있고 숫자가 너무 크지 않는 경우
- 장점? BigDecimal 보다 쓰기는 편하고 빠르다
- 단점? 다룰 수 있는 값의 크기가 제한되고 소수점을 직접 관리해야 한다

`int` 10진수 9자리, `long` 십진수 18자리, `BigDecimal` 는 그보다 큰 수도 표현 가능

Code

- 부동 소수점 리터럴은 double 형이다. float 타입을 표현하기 위해서는 숫자 끝에 F나 f 를 붙인다.

```
double delta = 453.523311903;
float ratio = 0.2363F;
```

- 소수점 계산에 double이나 float 을 쓰면 발생하는 문제

```
System.out.println(1.03 - 0.42);
```

```
System.out.println(1.00 - 9 * 0.1);
```

> 10센트, 20센트, 30센트, ... 1달러 짜리의 사탕을 사려고 한다. 내가 가진 돈이 1달러라고 한다면 가장 저렴한 사탕부터 산다고 했을 때 몇 개나 살 수 있을까? 또 남은 돈은 얼마가 될까?

1. 부동 소수 타입(double, float) 을 사용한 문제 풀이
2. BigDecimal 을 사용한 문제 풀이
3. 정수 타입(int, long) 을 사용한 문제 풀이

⚠ 부동 소수 타입

`funds` 내가 가진 돈 (단위: 달러)

`itemsBought` 구입한 사탕 갯수

`price` 사탕의 가격

```
public static void main(String[] args) {
    double funds = 1.00;
    int itemsBought = 0;
    for (double price = 0.10; funds >= price; price += 0.10) {
        funds -= price;
        itemsBought++;
    }
    System.out.println(itemsBought + " items bought.");
    System.out.println("Change: $" + funds);
}
```

남은 돈이 0.3999999999999999 로 잘못된 계산 값이 나온다.

🍀 BigDecimal

`funds` 내가 가진 돈 (단위: 달러)

`itemsBought` 구입한 사탕 갯수

`price` 사탕의 가격

```
public static void main(String[] args) {
    final BigDecimal TEN_CENTS = new BigDecimal(".10");

    int itemsBought = 0;
    BigDecimal funds = new BigDecimal("1.00");

    for (BigDecimal price = TEN_CENTS;
         funds.compareTo(price) >= 0;
         price = price.add(TEN_CENTS))
    {
        funds = funds.subtract(price);
        itemsBought++;
    }

    System.out.println(itemsBought + " items bought.");
    System.out.println("Money left over: $" + funds);
}
```

옳은 답이 나온다

🍀 정수 타입

BigDecimal 로 코딩하기 불편하고 성능이 떨어지는 점이 마음에 안 든다 😠 !

→ int, long 을 사용하는 것도 괜찮은 방법

(단, 표현할 수 있는 범위를 넘지는 않는지 고려해야 하고, 소수점을 직접 추적할 수 있어야 한다)

`funds` 내가 가진 돈 (단위: 센트)

`itemsBought` 구입한 사탕 갯수

`price` 사탕의 가격

```
public static void main(String[] args) {
    int itemsBought = 0;
    int funds = 100;
    for (int price = 10; funds >= price; price += 10) {
        funds -= price;
        itemsBought++;
    }

    System.out.println(itemsBought + " items bought.");
    System.out.println("Cash left over: " + funds + " cents");
}
```