Clean Code

▼ chap2 의미 있는 이름

- 의도를 분명히 밝혀라
- 그릇된 정보를 피하라

  예약어 사용x, 소문자 L, 대문자 O 사용 자제
- 의미 있게 구분하라
- 발음하기 쉬운 이름을 사용하라
- 검색하기 쉬운 이름을 사용하라

  자주 사용되는 변수나 상수는 검색하기 쉬운 이름(길고 겹치는 변수, 상수가 많이 없는 이름) 사용
- 인코딩을 피하라

  유형, 범위를 규칙을 정해 넣지 말기
- 자신의 기억력을 자랑하지 마라

  생략하지 말고 명료한 이름 사용
- 클래스 이름

  명사
- 메서드 이름

  동사
- 기발한 이름은 피하라
- 한 개념에 한 단어를 사용하라
- 말장난을 하지 마라

  한 단어를 두 목적으로 사용하지 마라.
- 해법 영역에서 가져온 이름을 사용하라

  개발 용어를 이름으로 활용해라(굳이 클라이언트의 용어를 사용할 필요가 없다).
- 문제 영역에서 가져온 이름을 사용하라

  적절한 프로그래머 용어가 없다면 사용한다.
- 의미 있는 맥락을 추가하라

의미있는 접두어를 추가하여 이름만 봐도 맥락을 알 수 있게 한다.

- 불필요한 맥락을 없애라

▼ chap3 함수

- 작게 만들어라

  블록과 들여쓰기를 많이 하지 마라.

  함수를 나눠서 기능을 구현하여 블록과 들여쓰기를 대체해라.

- 한 가지만 해라

  함수 내 섹션이 있다면 그를 함수로 더 쪼개라.

- 함수 당 추상화 수준은 하나로

  코드는 이야기처럼 위에서 아래로 읽혀야 좋다.

  위에서 아래로 내려가면서 함수 추상화 수준이 낮아져야 한다.

- Switch문

  최대한 쓰지 마라. polymorphism을 이용하여 최소한으로 써라.

- 서술적인 이름을 사용하라

- 함수 인수

  최대한 적게

  - 많이 쓰는 단항 형식
    - 질문을 던지는 경우: 입력을 결과로 출력을 반환
    - 이벤트: 입력만 있고 출력이 없음. 시스템 상태를 바꿈.
  - 플래그 인수

    함수로 bool 변수를 넘기는 경우.

    bool 값에 따라 다른 실행을 한다. → 한 함수가 여러가지를 한다는 뜻이므로 하지 말아야 한다.
  - 이항 함수
  - 삼항 함수
  - 인수 객체

    인수가 늘어나야 하면 클래스 변수로 선언하자.

변수를 묶어 넘기려면 이름을 붙여야 하므로 개념을 표현하게 된다.

    - 인수 목록

      인수 개수가 가변적인 함수는 단항, 이항, 삼항 함수로 취급할 수 있다.

    - 동사와 키워드

      단항 함수는 함수와 인수가 동사, 명사 쌍을 이뤄야 한다.

      함수 이름에 인수 이름을 넣으면 인수 순서를 기억할 필요가 없다.

- 부수 효과를 일으키지 마라

  출력 인수는 피해야 한다.

  함수에서 상태를 변경해야 한다면 함수가 속한 객체 상태를 변경하는 방식을 택한다.

- 명령과 조회를 분리하라
- 오류 코드보다 예외를 사용하라
    - Try/Catch 블록 뽑아내기
    - 오류 처리도 한가지 작업이다.
    - Error.java 의존성 자석

      오류 코드 대신 예외를 사용하면 의존성을 없앨 수 있다.

- 반복하지 마라
- 구조적 프로그래밍
    - 아주 큰 함수

      return 문은 하나여야 한다

      break나 continue를 사용하면 안된다.

      goto는 절대로 사용하면 안된다.

    - 작은 함수

      goto를 피해야 한다.

- 함수를 어떻게 짜죠?

  초안을 짜고 단위 테스트케이스를 만든다.

  코드를 다듬으면서 테스트케이스를 통과하는지 검사한다.