3장 함수

* 첫번 째 규칙 : 작게 만들어라
  + 함수는 한 화면이 넘어가면 안된다
  + 리펙토링을 계속하여 나눌수있을만큼 나누고 줄일 수 있을만큼 줄여야 한다.
* 두번 째 규칙 : 한가지만 해라
  + 함수는 한가지만 해야한다. 그 한가지를 잘해야 한다. 그 한가지만을 해야 한다.
  + 함수를 만드는 이유는 큰 개념을 다음 추상화 수준에서 여러 단계로 나눠 수행하기 위함
* 세번 째 규칙 : 함수 당 추상화 수준은 하나로
  + 함수가 확실히 한가지만 하려면 추상화 수준도 한가지로 해야한다.
* 네번째 규칙 : 스위치 문은 작게 만들자
* 다섯번 째 규칙 : 서술적인 이름을 사용하라
  + 이름 정하느라 시간을 들여도 괜찮다. 이런 저런 이름을 넣어 코드를 읽어보면 더 좋다.
  + 최대한 서술적인 이름을 사용하면 개발자 머릿속에서도 설계가 뚜렷해지므로 코드를 개선하기 쉬워진다.
  + 이름을 붙일 때 일관성이 있어야 한다.
* 여섯번쨰 규칙 : 함수 인수
  + 이상적인 함수의 인수는 0개이다. 다음은1, 다음은2, …
  + 인수는 함수의 개념을 어렵게 만든다.
  + 최선은 입력 인수가 없다는 경우이고 차선은 입력 인수가 1개뿐인 경우이다.
* 많이쓰는 단항형식
  + 인수에 질문을 던지는 경우
  + 인수를 뭔가로 변환해 결과를 반환하는 경우
* 플래그 인수
  + 플래그인수는 부울 값을 넘기는 관례로 매우 끔찍하다. 함수가 한꺼번에 여러가지를 처리한다고 공표한 셈이므로
* 부수효과를 일으키지 마라
  + 부수표과는 거짓말일 뿐 한가지를 하겠다고 약속하고선 다른 것도 수행함
* 출력인수
* 명령과 조회를 분리하라
  + 명령의 기능과 조회의 기능을 따로 분리하여 혼란을 뿌리뽕으란느 것
* 오류코드보다 예외를 사용하라
* 반복하지 마라
  + 기본중의 기본, 반복은 함수에서 항상 없어져야 한다
* 구조적 프로그래밍
* 함수를 짜는 법
  + 글짓기와 마찬가지로 먼저 함수를 길게 짠다 생각할수록
  + 그리고 해당하는 함수의 반복과 기능적인 측면에서 함수를 줄인다.
  + 두번 세번 확인하면서 다듬는다.