알고리즘 강의 소개

알고리즘

- 주어진 문제를 해결하는 다양한 문제해결 **알고리즘** 공부 중요
- 알고리즘
    - 주어진 문제의 모든 사례에 대해 동일하게 작동하는 단계별 명령의 목록, 즉 말해 컴퓨터 프로그램을 가리킴
- 계산 가능computable한 문제: 해결 알고리즘이 존재하는 문제

알고리즘의 다양성

- 문제 하나에 대해 다양한 알고리즘 존재 가능
- 알고리즘의 복잡도에 따라 실행 방식이 다를 수 있음.
- 다양한 알고리즘을 비교, 분석할 수 있어야 함

계산 불가능한 문제

- 컴퓨터로 계산이 불가능한 문제도 존재
- 예제: 튜링의 정지 문제
- 매우 어려운 주제임

자료 구조

- 하나의 알고리즘을 여러 종류의 프로그래밍언어로 구현 가능
- 프로그래밍언어의 특성에 따라 구현 방식과 성능이 달라질 수 있음
- 알고리즘 구현 필수 요소
    - 자료 구조: 문제의 사례 표현
    - 명령문: 알고리즘의 작동 단계 묘사

- 기본 자료 구조: 대부분의 프로그래밍언어가 제공
    - 정수
    - 문자열
    - 부동소수점
    - 부울값
- 기본 명령문
    - 변수 할당
    - 조건문: `if...else...` 등
    - 반복문: `for`, `while` 등

코드 추상화와 고성능 자료 구조

- 원칙적으로는 기본 자료 구조와 기본 명령문만을 이용하여 모든 계산 가능한 문제 해결 가능
- 코드 추상화와 고성능 자료 구조: 복잡한 문제를 보다 효율적인 알고리즘으로 해결하기 위해 필요

코드 추상화

- 다양한 코드 추상화 활용됨
- 여기서는 절차 추상화와 데이터 추상화 두 종류만 언급

절차 추상화

- 알고리즘: 적절한 **API**(Application Programming Interface)들의 조합
- 여기서는 API의 구현 자체는 관심 대상 아님

절차 추상화 예제

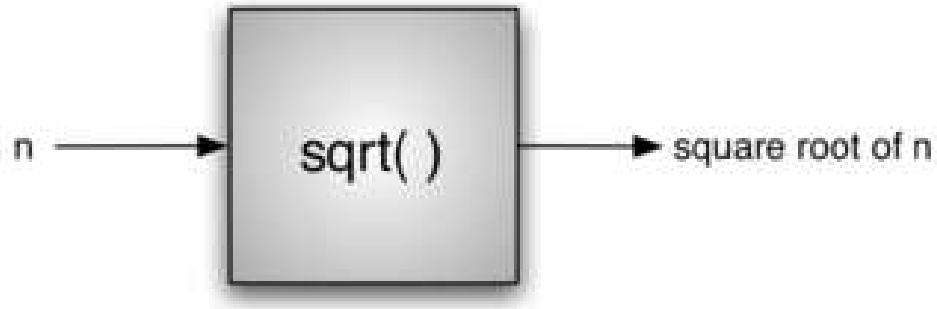
- 예제: 제곱근 계산 함수
    - `sqrt()`는 `math` 모듈에 정의되어 있음
    - 사용법은 다음과 같음

In [1]:
```
import math

math.sqrt(16)
```

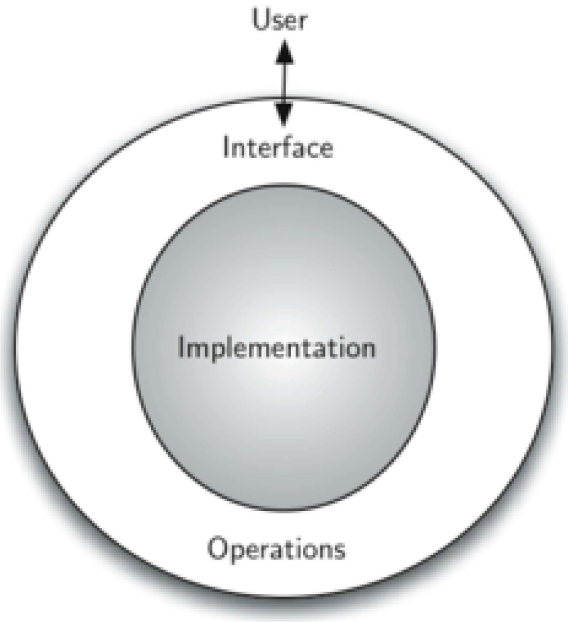
Out[1]: 4.0

- `sqrt()` 함수가 제곱근을 계산하는 함수라는 사실만 중요.
- 즉, 주어진 API의 기능과 사용법을 아는 게 중요



데이터 추상화

- 데이터 추상화: 추상 자료형을 이용하여 적절한 자료 구조를 정의하는 기법
- 추상 자료형Abstract data type(ADT): 특정 종류의 데이터와 그 데이터와 관련된 API들을 하나로 묶어 다루는 방법 묘사



- 사용자의 입장에서 추상 자료형을 어떻게 직접 구현했는가는 중요하지 않음
- 해당 추상 자료형을 구현한 객체가 제공하는 인터페이스를 통해 해당 객체를 활용할 수 있어야 함.
- 데이터 캡슐화: 추상 자료형이 데이터와 함께 관련 API를 하나로 묶어 처리한다는 의미

자료 구조

- 추상 자료형을 클래스, 구조체 등 많은 기능을 제공하는 프로그래밍 요소로 구현한 대상

- 추상 자료형과 자료 구조를 분리해서 다루는 이유: 데이터 추상화를 이용하면 다루고자 하는 데이터의 기본 속성과 사용법의 핵심을 보다 쉽게 전달할 수 있기 때문
- 문제를 해결하는 알고리즘을 구현하려면 적절한 추상 자료형을 구체적인 자료 구조로 구현해야 함.

파이썬 프로그래밍언어

- 알고리즘을 가장 자연스럽게 표현해주는 언어 중에 하나임
- 클래스를 이용하여 자료 구조 구현
- 객체 지향 프로그래밍 기법 적용

프로그래밍 환경

- 구글 코랩 기본 사용법
- 주피터 노트북 기본 사용법과 추가 설정

전제 사항

- 파이썬 프로그래밍 기초