

빅 데이터 혁신 공유 대학

# 파이썬으로 배우는 데이터 구조

---

한동대학교 전산전자공학부

김영섭 교수



교육부



한국연구재단



# Data Structures in Python

## Chapter 2 - 1

- JavaScript Object Notation(JSON)
- **Software Design Principles**
- Abstract Data Type(ADT)

나는 인애를 원하고 제사를 원하지 아니하며 번제보다 하나님을 아는 것을 원하노라 (호6:6)  
하나님은 모든 사람이 구원을 받으며 진리를 아는데에 이르기를 원하시느니라 (딤후2:4)

그런즉 너희가 먹든지 마시든지 무엇을 하든지 다 하나님의 영광을 위하여 하라 (고전10:31)

# Agenda & Readings

---

- **Some Software Design Principles**
- Abstract Data Type (ADT)
  - What is an ADT?
  - What is a Data Structure?
- Examples on ADT:
  - Integer, Set, and List
- Resources:
  - Textbook: [Problem Solving with Algorithms and Data Structures](#)
    - Chapter 1.5 [Why Study Data Structures and Abstract Data Types?](#)
  - Textbook: [www.github.com/idebtor/DSpy](http://www.github.com/idebtor/DSpy):
    - [Ch1-1: Introduction](#)

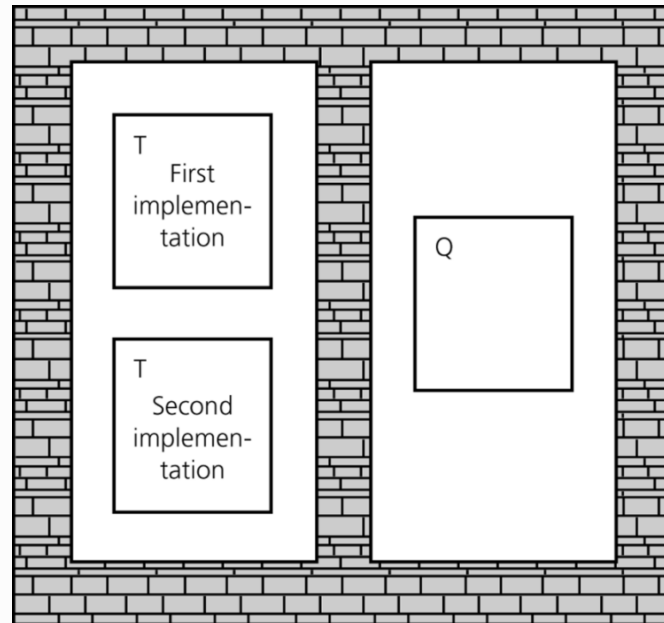
# Software Design Principles - Modularity

---

- Modularity divides a program into manageable parts.
  - Keeps the complexity of a large program manageable.
  - Isolates errors.
  - Eliminates redundancies.
  - Encourages reuse (write libraries).
- A modular program is
  - Easier to write.
  - Easier to read.
  - Easier to modify (or maintain).

# Software Design Principles - Information Hiding

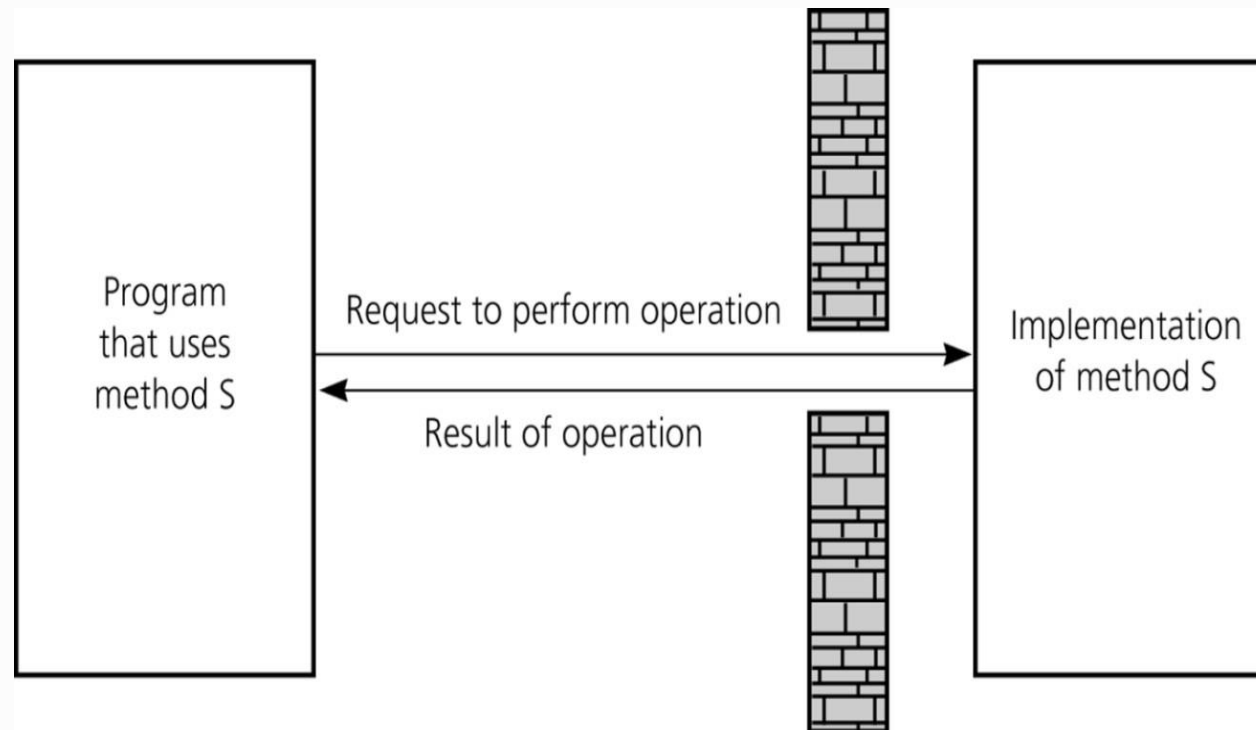
- Hides certain implementation details within a module.
- Makes these details inaccessible from outside the module.
- Isolates the implementation details of a module from other modules.



**Isolated tasks:** the implementation of task T does **not** affect task Q

# Software Design Principles - Isolation of Modules

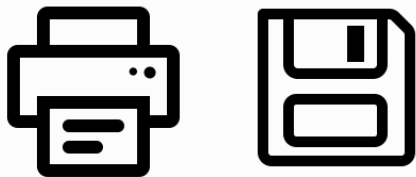
- The isolation of modules is not the end. (otherwise module would be useless)
- Methods' specifications, or contracts, govern how they interact with each other.
- Similar to having a wall hiding details, but being able to access through “hole in the wall”



# Software Design Principles - Abstraction

---

- What does 'abstract' mean?
  - From Latin: to 'pull out'—the essentials
  - To defer or hide the details
  - Abstraction emphasizes **essentials** and defers the details, making engineering artifacts easier to use.
- Example:
  - I don't need a mechanic's understanding of what's under a car's hood in order to drive it.
    - What's the car's interface?
    - What's the implementation?





# Software Design Principles - Abstraction

---

- Abstraction
  - The principle of ignoring those aspects of a subject that are not relevant to the current purpose in order to concentrate solely on those aspects which are **relevant**.
- How do we achieve:
  - Modularity
  - Information hiding
  - Isolation of modules
    - i.e. The abstraction of **what** from the **how**

# Software Design Principles - Abstraction

---

- Abstraction separates the purpose and use of a module **from its implementation**.
- A module's specifications should
  - Detail how the module behaves.
  - Identify details that can be hidden within the module.
- Advantage:
  - Hides details (easier to use).
  - Allows us to think about the general framework overall solution) & postpone details for later.
  - Can easily replace implementations by better ones and it **does not affect the users of the modules (or library)**.

# Software Design Principles - Data and Operations

---

- What do we need in order to achieve the above Software Design principles?
- **For example**, what are the typical operations on data? (example: database)
  - Add data to the database
  - Remove data from the database
  - Find data (or determine that it is not in the data base)
  - Ask questions about the data in a data collection  
(e.g. how many CS50 students take Linear Algebra course?)
- **Question:** Do we need to know what data structures used?
  - No, better make implementation independent of it!

# Abstraction - Example

- Programming Language
  - Programming:
    - For instructing a computer to perform a computing task
    - But more than just a means of computing ...
      - a framework within which we **organize our ideas**
      - a means of **communicating ideas** in the community
  - Language:
    - Provides for combining **simple** ideas to form more **complex** ideas.
- Example

羊 手 戈



義

[sheep]

[hand]

[spear head]

[righteousness]

손(손 수手)으로 양(羊)를 쳐서(창 과戈) 드림.

손의 창으로 양을 잡아 그 피로 나를 덮는 것이 옳을 의  
손의 창으로 나를 쳐서 어린 양 예수님께 복종하는 것이 옳은 의

# Abstraction - Example

- Programming Language
  - Programming:
    - For instructing a computer to perform a computing task
    - But more than just a means of computing ...
      - a framework within which we **organize our ideas**
      - a means of **communicating ideas** in the community
  - Language:
    - Provides for combining **simple** ideas to form more **complex** ideas.

- Example

牛 羊 秀 戈 ⇒ 犧

[ox] [sheep] [excellent] [spear head] [sacrifice]

흠 없는(秀) 양(羊)과 소(牛)를 창(칼)로 잡아(戈) 희생시킴으로써(犧) 하나님께 제물로 드렸다 (레1:2-4)

# Software Design Principles - Data and Operations

- Asks you to think what you can do to a collection of data independently of how you do it.
- Allows you to develop each data structure in relative isolation from the rest of the solution.
- A natural extension of procedural abstraction.



```
CAR:  
Start()  
Stop()  
TurnLeft()  
TurnRight()
```

```
CAR:  
Start(){  
    Select first gear  
    Release parking brake  
    Bring clutch up to the friction point  
    Press gas pedal  
    Release clutch  
}
```

NOTE: Implementation can be different for different cars, e.g. automatic transmission.

# Data Structures in Python

## Chapter 2 - 1

- JavaScript Object Notation(JSON)
- **Software Design Principles**
- Abstract Data Type(ADT)