빅 데 이 터 혁 신 공 유 대 학

파이썬으로 배우는 데이터 구조

한동대학교 전산전자공학부

김영섭 교수

교육부 NRF 한국연구재단

Data Structures in Python
Chapter 2 - 1

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

Prof. Youngsup Kim, idebtor@gmail.com, CSEE Dept., Grace School Rm204 Handong Global University

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:
    - Ch1-1: Introduction

Abstract Data Types

- An Abstract Data Types (ADT) is composed of
  - A collection of **data**
  - A set of **operations** on that data
- Specifications of an ADT indicate **what** the ADT operations do, **not how** to implement them.
- Implementation of an ADT includes choosing a particular data structure. This let the coders enjoy the freedom of choices in terms of how they code.

Abstract Data Types - Properties and Behaviors

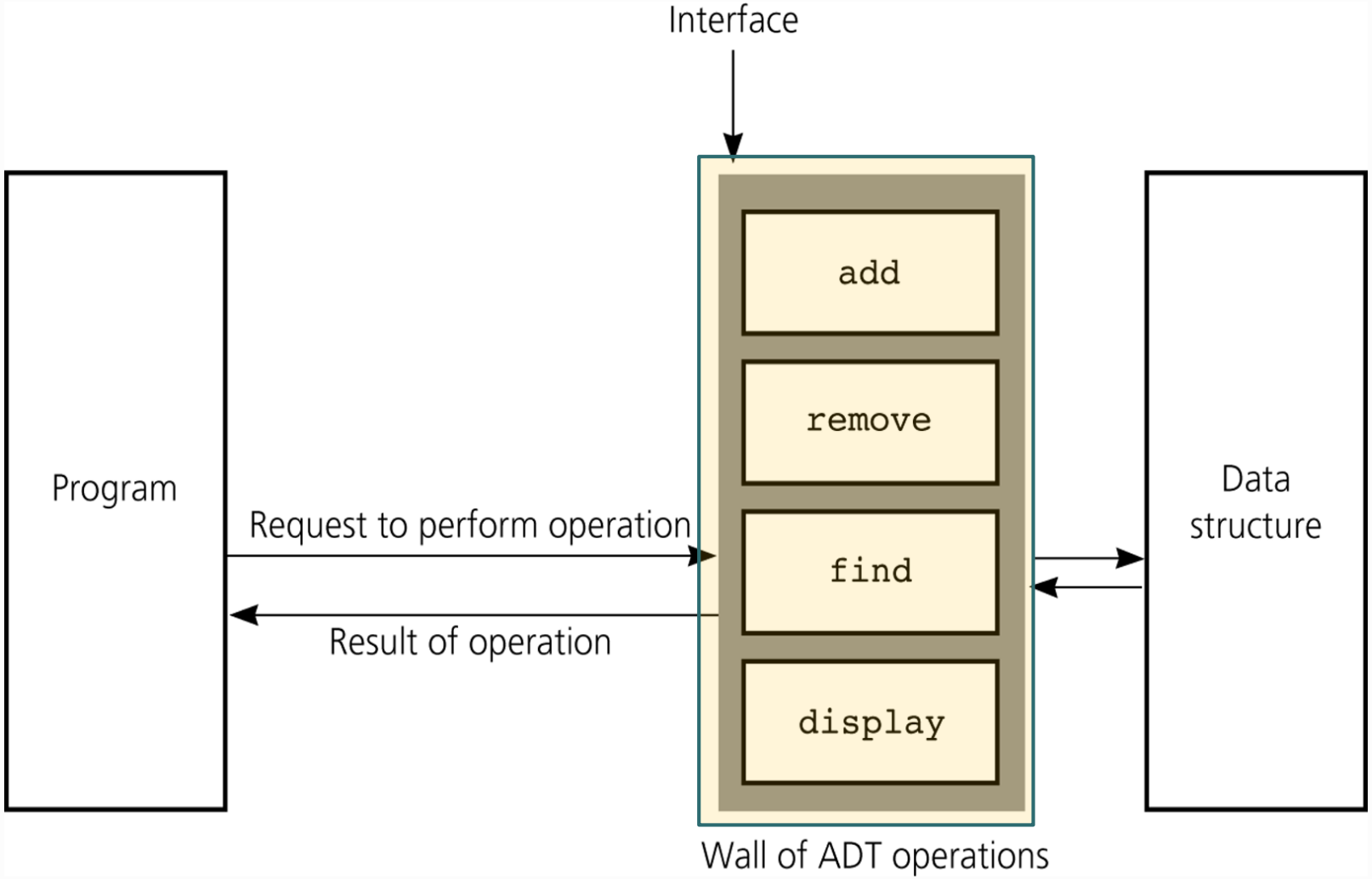
- **The properties** of the ADT are described by the collection of data.
  - The data can be in terms of simple data types and/or complex data types.
  - Simple data types
    - Integer
    - Floating point
    - Character
  - Complex data types
    - Multimedia
- **The behaviors** of the ADT are its operations or functions.

Abstract Data Types - Data structure vs. ADT

- An ADT is not the same with a Data Structure
- Data structure
  - A construct that is defined within a programming language to store a collection of data.
  - Example: arrays
- ADT provides “data abstraction”
  - Results in a wall of ADT operations between data structures and the program that accesses the data within these data structures.
  - Results in the data structure being hidden.
  - Can access data using ADT operations.
  - Can change data structure without changing functionality of methods accessing it.

Abstract Data Types

- A wall of ADT operations isolates a data structure from the program that uses it.



Abstract Data Types - Disadvantages & Advantages

- Disadvantages of Using ADTs
  - Initially, there is more to consider
    - Design issues
    - Code to write and maintain
    - Overhead of calling a method to access ADT information
    - Greater initial time investment
- Advantages of Using ADTs
  - A client (the application using the ADT) doesn't need to know about the implementation.
  - Maintenance of the application is easier.
  - The coder can focus on problem solving and not worry about the implementation.

ADT Examples - Designing an ADT

- An abstract data type (ADT) is a specification of a set of **data** and the set of **operations** that can be performed on the data.
- Such a data type is **abstract** in the sense that it is **independent** of various concrete implementations.
  - Questions to ask when designing an ADT
    - What data does a problem require?
    - What operations does a problem require?
  - Examples:
    - Integers, floating-point
    - Sets
    - Lists
    - Stacks, Queues, Trees …

ADT Examples - Integers

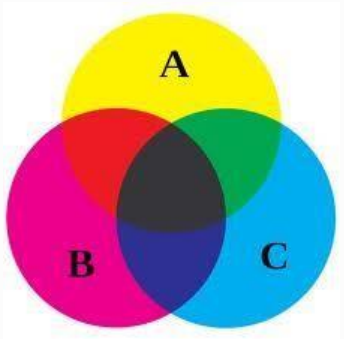
- Data
  - Containing the positive and negative whole numbers and 0
- Operations which manipulate the data:
  - Such as addition, subtraction, multiplication, equality comparison, and order comparison
- Methods for data conversion, output etc.

ADT Examples - Floating-point

- Data
  - Containing the whole possible positive and negative values
  - Precision is limited by number of digits
  - Arrays, strings, lists…
- Operations which manipulate the data:
  - Such as addition, subtraction, multiplication, equality comparison, and order comparison
  - Complex mathematical functions such as exponential, logarithm, triangular functions etc.
  - Rounding's
  - Methods for data conversion, output etc.

ADT Examples - Sets

- Data
  - An unordered collection of unique elements
- Operations which manipulate the data:
  - add
    - Add an element to a set
  - remove
    - Remove an element from the set,
  - Others
    - Get the union, intersection, complement of two Set objects

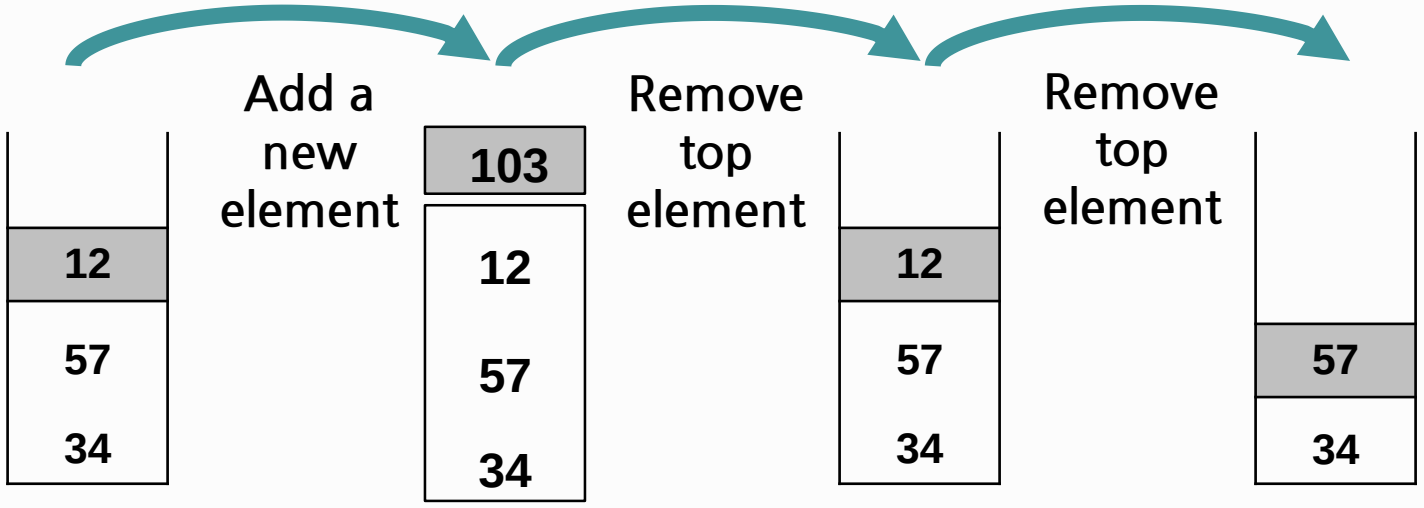


ADT Examples - Linear Structures

- **Linear structures** are data collections whose items are ordered depending on how they are added or removed from the structure.
- Once an item is added, it stays in that position relative to the other elements that came before and came after it.
- Linear structures can be thought of as having two **ends**, **top** and **bottom**, (or front and end or front and back)
- What distinguishes one linear structure from another is the way in which items are added and removed, in particular the location where these additions and removals occur, e.g., add only to one end, add to both, etc.

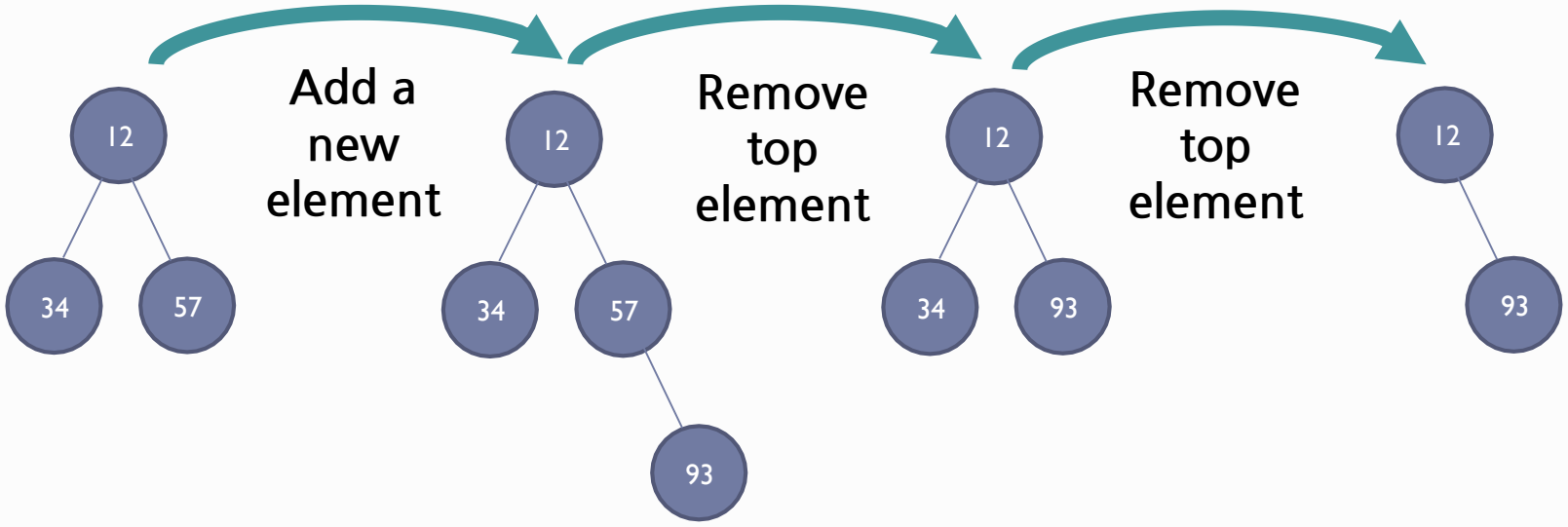
ADT Examples - Linear Structures

- Examples: Stack, Queue, Linked-list, etc.



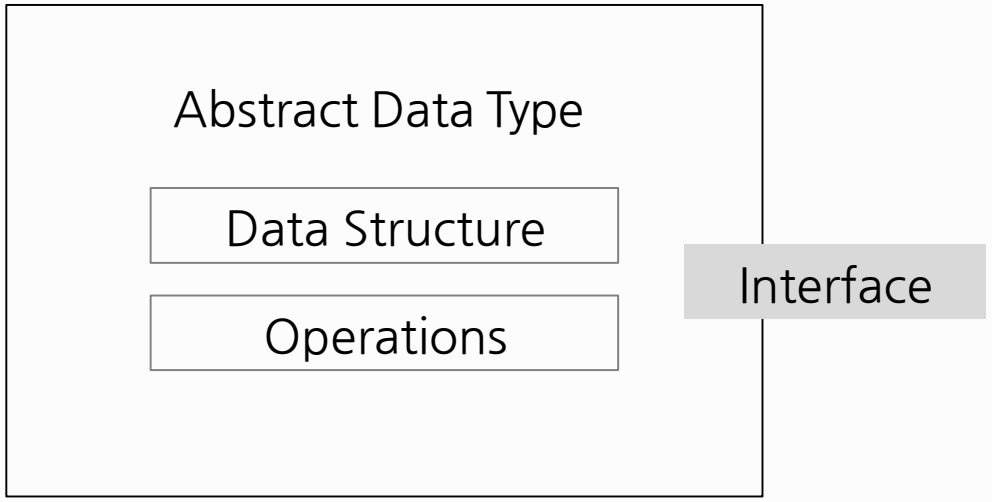
ADT Examples - Non-Linear Structures

- Every data item is attached to several other data items in a way that is specific for reflecting relationships.
- The data items are not arranged in a sequential structure
- Examples: Tree, Graph, Heap, etc.



Summary

- Solving a problem becomes easier by considering only relevant data and operations and ignoring the implementations (“abstraction”)
- Abstract Data Types (ADTs) enable you to think -
  - **What** you can do with the data independently of **how** you do it
- An abstract data type can be accessed by a limited number of **public methods** -
  - They are often defined by using an interface
- The implementation of the ADT (data structures and algorithms) can be changed without influencing the behavior of the ADT.



Data Structures in Python
Chapter 2 - 1

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

Prof. Youngsup Kim, idebtor@gmail.com, CSEE Dept., Grace School Rm204 Handong Global University