Abstract Data Types

CMPT 145

Data Types

- A data type is:
  - A set of values
  - Operations that can be performed on those values.
- Some data types are provided by a language.
- New data types can be added to any language!

Data Type Examples

- Integers
  - Values: $\{..., -2, -1, 0, 1, 2, ...\}$
    - In Python, the integer data type has infinite range of values.
    - In many other languages, integers have finite range!
  - Operations:
    - Addition ($+$)
    - Multiplication ($*$)
    - Modulo Division ($\%$)
    - (and many more...)

Data Type Examples

- Booleans
  - Values: {True, False}
  - Operations:
    - OR
    - AND
    - NOT
- Some (old) languages do not have a Boolean data type, so they pretend that 0 means $False$ and any other number is $True$)

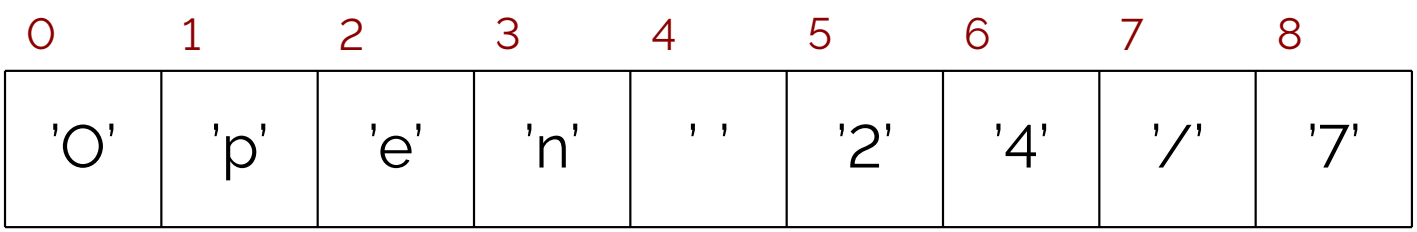
Data Structures

A data structure is a data type whose values are compound values, organized in some way.

- Compound means multiple values are grouped together as part of a single collection.
- The organization gives the data type its structure.
- A data structure also has a set of operations on its values.

Data Structures Example

- Strings
  - Values: The set of all character sequences.
    - e.g. `"dog"`, `"Open 24/7"`
  - Operations:
    - `len()`
    - Method `isdigit()`
    - Method `find()`
    - Method `replace()`



Data Structures Examples

- Lists
  - Values: The set of all value sequences (any kind of value!)
    - e.g. `["behemoth",10,100]`
  - Operations:
    - `len()`
    - Method `append()`
    - Method `insert()`
    - Method `reverse()`

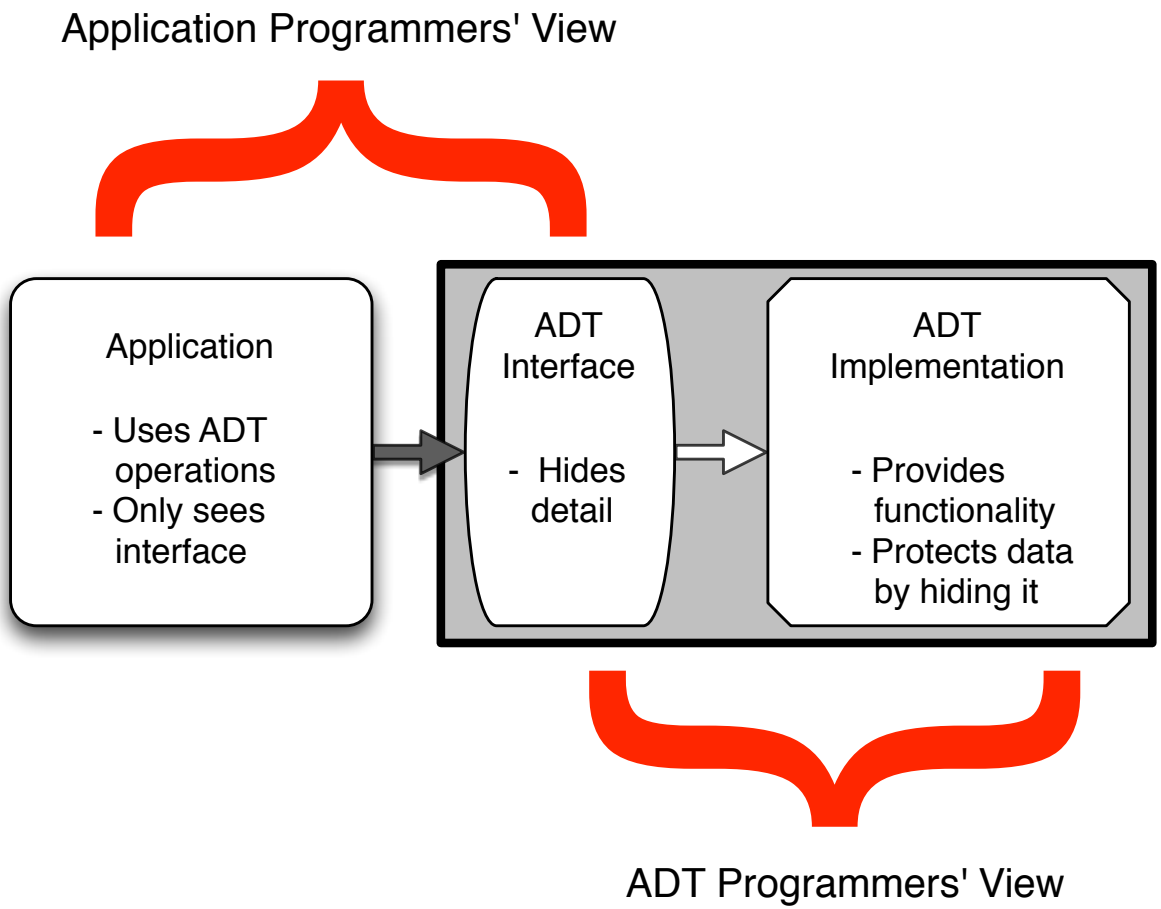
Other Data Structure Examples

- Tuples
- Complex numbers
- Student records, employee records, Guinness World Book Records, …
- A significant proportion of software design is choosing or designing data structures!

Abstract Data Types

- An Abstract Data Type (ADT) is a data type that
  1. Hides the data values.
     Can only change values through its operations
  2. Hides the operation implementation.
     Only the interface is known.
  3. Hides the details about the organizational structure.
     Using the operations means you do not need to know the details!
- Described in terms of their:
  1. Purpose
  2. Operations
  3. Organization

How ADTs are used



ADT Examples: Python Lists

- Purpose:
  - Manage linear sequences of data values.
- Implementations:
  - Data:
    - Linked Nodes
    - Fixed Length Data Arrays
  - Operations:
    - Access a value by index
    - Remove, append, insert, reverse, …

ADT Examples: Python Dictionary

- Purpose:
  - Manage pairs of associated data items
- Implementations:
  - Data:
    - Hash Tables
    - Binary Search Trees
  - Operations:
    - Look up a value by key
    - Add/Remove a key-value pair

ADT Example: Registry

- Purpose:
  - Manage a collection of indexed Boolean values
- Implementation:
  - Data structure:
    - List of Booleans
  - Operations:
    - Create a registry
    - Set a value to True
    - Set a value to False
    - Check a given value

Demo 1

Use the Registry ADT to improve our code solution to the Sieve of Eratosthenes problem

- How does the ADT support our design goals?
- How does the ADT support our implementation goals?

Demo 2

Use the Registry ADT to improve our code solution to the Coupon Collector problem

- How does the ADT support our design goals?
- How does the ADT support our implementation goals?

Statistics ADT

- Purpose:
  - Computes statistics of a numerical data set
- Implementation:
  - Data Structure:
    - Record (dictionary)
  - Operations:
    - Create an empty data set
    - Add number to data set
    - Compute statistic

Demo 3

Use the Statistics ADT to enhance our code solution to the Gambler's Ruin problem

- How does the ADT support our design goals?
- How does the ADT support our implementation goals?