Abstract Data Types (ADT)

- Just a jargon for a data type:
  - That has a name
  - That has possible values (state)
  - Of which several instances can be created
  - That has operations on values/state

- e.g.: int → name
  - 0, 1, -1, 2, -2 ... → values
  - int x; → creating instances
  - +, -, *, /, Math.abs → operations

- e.g.: String → name
  - "", "Hello" ... → values
  - String n; → instance
  - +, substring, charAt, ... → operations
Tools & reuse

- How to solve
  - a problem
  - given some tools

- Options:
  - You could map problem into tools available
    - e.g., build a home using hammer, saw, screw driver
  - Or make new tools that are better for the problem
    - Power saw, template, nailer, ...

- If you have a lot of houses to build, second option is better
  - Same is true for programming...

Java programming tools

- Tools provided by Java:
  - Data types: int, double, char, boolean, String
  - Operations: +, -, *, /, ...
  - Control statements: if, for, function, ...

- Consider the problem of representing days of week
  - Could use: 0 for Sunday, 1 for Monday, ..., 6 for Saturday
    ```java
    int day;
    if (day == 1) // Monday
        ...
    if (day >=1 && day <= 5) // weekday
        ...
    ...
    ``
  - Ø: hard to read, creates new vocabulary
    "Did you take 3 off?" "No, I worked 3 and was off 4."
Better way?

- **Use mnemonic constants**
  ```java
  public final int SUNDAY = 0;
  public final int MONDAY = 1;
  ...
  public final int SATURDAY = 6;
  ...
  if (day == MONDAY)
  ...
  for (int d=MONDAY; d <= FRIDAY; d++) // weekdays
  ...
  » ⊗: more readable, but tedious to do
  public final int ACE_OF_DIAMONDS = 1;
  public final int TWO_OF_DIAMONDS = 2;
  ...
  ```

Object oriented languages

- **Create new tools in a way that facilitates re-use**
- **Allow us to create new data types (class)**
  - Name
  - Values
  - Instances (objects)
  - Operations (subroutines/methods)
The object model

- Example: payroll program needs employee info
  - Id, Fname, Lname, Adr, ...

- How to store?
  - Companion arrays
    - 😊: works
    - 😞: data not unified
      - e.g., swap first 2
        - many swaps
  - Table (multi-dim array)
    - 😊: works in JS, but not Java
      - All entries must have same type
    - 😞: relation between related elements is weak
      - e.g., sort on Iname and forget to include other columns
        - Like an Excel spreadsheet

Abstract Data Types

- We can create our own data type which has:
  - Name, values, instances, operations

- Class: template/prototype
  - Specifies data (state)
  - Specifies operations (usually to set/retrieve state)
  - Can create instances -> Objects

- e.g., create an Employee class
  - Can create several instances (objects) of Employee class

- 😊:
  - All same type: Employee
  - Can have a homogeneous array (of type Employee)
  - Helps treat each employee as a unit (single entity)
    - Safely swap (with 3 lines of code)
Questions

- How to define a class (template)?
- How to create objects of that class?
- How to read data from object?
- How to write values to objects?
- How to call methods in objects?
- How to initialize an object?

- We’ll address each of these
  - Running example: a stripped down version of Employee
    - name
    - htlInch (height in inches)
  - We’ll add a little to it later …