Abstraction

Dr. Mattox Beckman

Illinois Institute of Technology
Department of Computer Science

Objectives

- Explain the concept of abstraction and abstract data-type.
- Describe the components of an abstraction.

Motivating Example

Suppose you are given a pile of exams. You need to organize them. How do you do it?

Main Choice: How to Sort

The kind of questions you want to ask determine the organization you use.

- "What did Mr. X get on his exam?"
- "What were the top 3 scores?"
- "Give me all the exams of people who got a C."
- "How did Ms. Y do on this exam compared to her last exam?"
- "Did the best 3 international students do better than the best 3 domestic students?"

Question: What happens if you guess wrong?

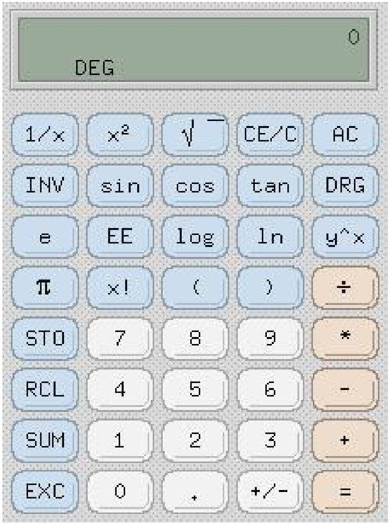
Components of a Model

- First, you need a *phenomenon* to model.
- Second, you ask some questions about the phenomenon.
  - Question 1: What are the elements of the phenomenon?
  - Question 2: What are the properties of the elements?
  - Question 3: Which of these elements and actions do we care about?
  - This is the *interface* or *view*.
- Third, you simulate the phenomenon.
  - Create an *implementation* for your model.
  - Use the interface to restrict access to the model.
- Important question: what are the limitations of the model?

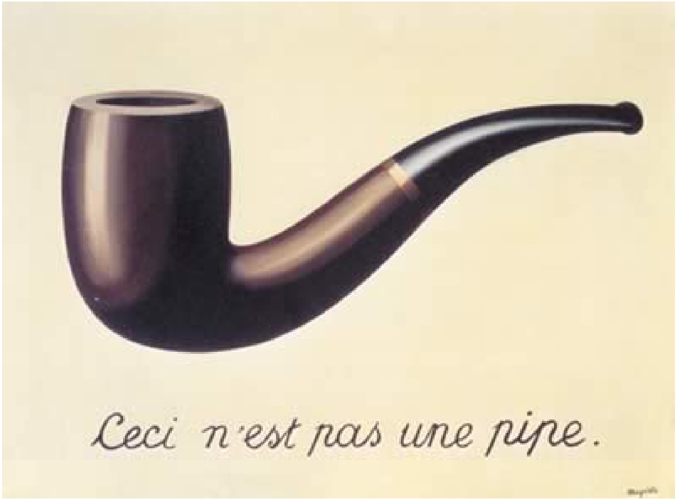
Example 1: Integers.

- Phenomenon: the integers.
- Questions:
  - Elements: zero, and positive and negative whole numbers.
  - Properties: infinite set, can add, increment, subtract, etc.
  - Don't care: maybe it's not that important to count to ∞.
- Simulation: use 16 *binary digits*. First one is a sign bit.
  - Use built-in assembly operations to manipulate the bits.
- Limitations: can't count past 32767.

Example 2: Calculator



Example 3: Pipe



Motivation

Here is some code from the Linux kernel.

```
char * strcat(char * dest, const char * src)
{
        char *tmp = dest;
        while (*dest)
                dest++;
        while ((*dest++ = *src++) != '\0');
        return tmp;
}
```

- What is a string? (What are the elements? ... actions?)
- How do `dest` and `src` act like strings?
- How do they **not** act like strings?

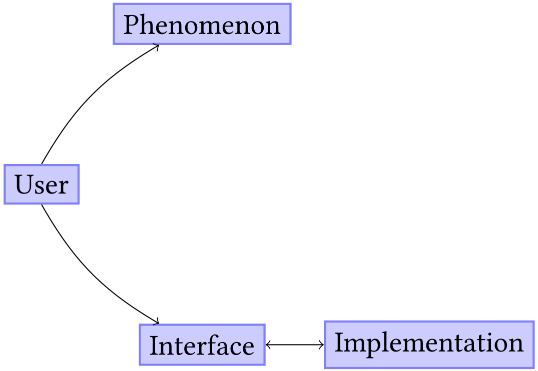
Abstract Data Type

“A type whose internal form is hidden behind a set of access functions. Objects of the type are created and inspected only by calls to the access functions. This allows the implementation of the type to be changed without requiring any changes outside the module in which it is defined.”

- internal representation
- encapsulation / hidden
- access via “functions”

An ADT is a contract between the objects and the rest of the program.

Picture



- From the user's perspective, interaction with the phenomenon and interaction with the model (via the interface) is indistinguishable.
- The implementation can be changed at will without affecting the user's experience.

Other Thoughts

- An implementation itself can be thought of as a phenomenon.
  - E.g., using integers to model booleans values… or states in a machine…
- Computer Programming is very much like Wizardry.

“Any technology, sufficiently advanced,
is indistinguishable from magic.”
— Isaac Asimov