Abstraction

The Key to Managing Complex Processes

Notes for CSC 100 - The Beauty and Joy of Computing
The University of North Carolina at Greensboro

Class Exercise

In groups of 3-5 students:

Make a list of steps you take in the morning from waking up to being ready to go to school or work.

Obviously everyone might do things a little differently, but come up with a sequence of steps you can all agree on.

Forms of Abstraction

Descriptions and Example from Dan Garcia, UC Berkeley

- Detail removal
  "The act or process of leaving out of consideration one or more properties of a complex object so as to attend to others."

- Generalization
  "The process of formulating general concepts by abstracting common properties of instances."

From Dan Garcia, UC Berkeley

Henri Matisse "Naked Blue IV"
**Question:** What is this?

![Image of a Dell Inspiron Desktop](image)

**Detail Removal Example**

*Possible answers to previous question*

A detailed answer:

A Dell Inspiron Desktop, model i620-1996BK, with a 3.3 GHz Intel i3 processor, 4 GB of RAM, 500 GB 7200 rpm hard disk, Intel HD Graphics 2000, USB optical mouse, and pre-installed with Windows 7 Home Premium (64 bit).

Just a few important technical details:

A Dell Inspiron Desktop with 4 GB of RAM and 500 GB hard disk.

The most basic description:

A computer.

**Important point:** Different levels of detail are suitable in different situations. An office designer doesn't need to think of this as anything other than "a computer" that needs to be placed in the room - details are superfluous and distract from what the designer is trying to do!

---

**Detail Removal**

*A programming example*

A program exists in many different levels of detail:

A high-level language (e.g., C++)

\[ x = (y + 4) \times z - 3 \]

Assembly language (readable but detailed):

- movl -4(%ebp), %eax
- addl $4, %eax
- imull -8(%ebp), %eax
- subl $3, %eax
- movl %eax, -12(%ebp)

Machine language (what is really executed):

```
01000101 11111100 10000011 11000000
00000100
00001111 10101111 01000101 01000101
1000101
10101111 00110010 01000000
11100100
```

**Question:** If automated tools do this translation, why are multiple levels of abstraction useful?
Detail Removal

**BYOB Example**

This: 🔸 🔴 🔵 🔡

Really does something like this:

(And even that is simplified...)

Are the blocks provided by BYOB the only abstractions you will ever need?

NO! In this week's lab we'll see how to define our own blocks to make our own abstractions!

---

Generalization Example

- You have a farm with many kinds of animals
- Different food for each
- You have directions that say
  - To feed dog, put dog food in dog dish
  - To feed chicken, put chicken food in chicken dish
  - To feed rabbit, put rabbit food in rabbit dish
  - ...
- How could you do better?
  - To feed <animal>, put <animal> food in <animal> dish

From Dan Garcia, UC Berkeley

---

Generalization in Programming

**BYOB example**

Think about this block: 🔸 🔴 🔵 🔡

BYOB could have provided a block that just pointed up...
- and one that just pointed down...
- and one that just pointed right...
- and one that just pointed left...

Instead have one generalized block, which is
- easy to think about and use,
- less worry after the initial development effort, and
- more powerful (can point at any angle).