



The Beauty and Joy of Computing

Lecture #3 : Creativity & Abstraction



UC Berkeley EECS
Sr Lecturer SOE
Dan Garcia



LEARN LANGUAGE FREE!

Luis von Ahn's recent project is Duolingo, which is simultaneously allowing its users to learn a second language free, while also providing a cost-effective way to get documents translated on the web.

PREVENTING ADDICTION!

Researchers in the UK said that some gamers play up to 90 hrs a session, developing a "pathological" addiction. They say game makers need to do more, to get "sensible gaming"

www.duolingo.com

www.bbc.co.uk/news/uk-wales-south-east-wales-23576035



Computing is a Creative Activity

- “Creativity and computing are **prominent forces in innovation**; the innovations enabled by computing have had and will continue to have **far-reaching impact**.”
- At the same time, **computing facilitates exploration** and the creation of knowledge.
- This course will emphasize these **creative aspects of computing**.





Computing enables people...

- ...to translate intention into **computational artifacts**.
- A computational artifact is created by human conception using software tools.
- Examples of computational artifacts include
 - digital music, videos, images
 - documents
 - combinations of these. E.g.,
 - infographics
 - presentations
 - web pages.





Computing enables people...

- ...to **create** digitally!
- **Creating...**
 - knowledge
 - tools
 - expressions of ideas
 - solutions to problems.
- **Creating digitally...**
 - requires understanding and using software tools.
 - can be done by...
 - combining and modifying existing artifacts
 - creating new artifacts.





Collaboration is an essential part...

- ...of creating computational artifacts.
- Collaboration facilitates **multiple perspectives** in developing computational artifacts.
- A computational artifact can reflect collaborative intent.





We can **analyze** computational artifacts...

- ...for correctness, functionality, and suitability.
- A computational artifact may have weaknesses, mistakes, or errors depending on the type of artifact.
 - For example, music created by a program may not have an error but may simply be hard to listen to.
- The **functionality** and **suitability** (or **appropriateness**) of a computational artifact may be related to **how it is used** or **perceived**.





Computing extends traditional forms...

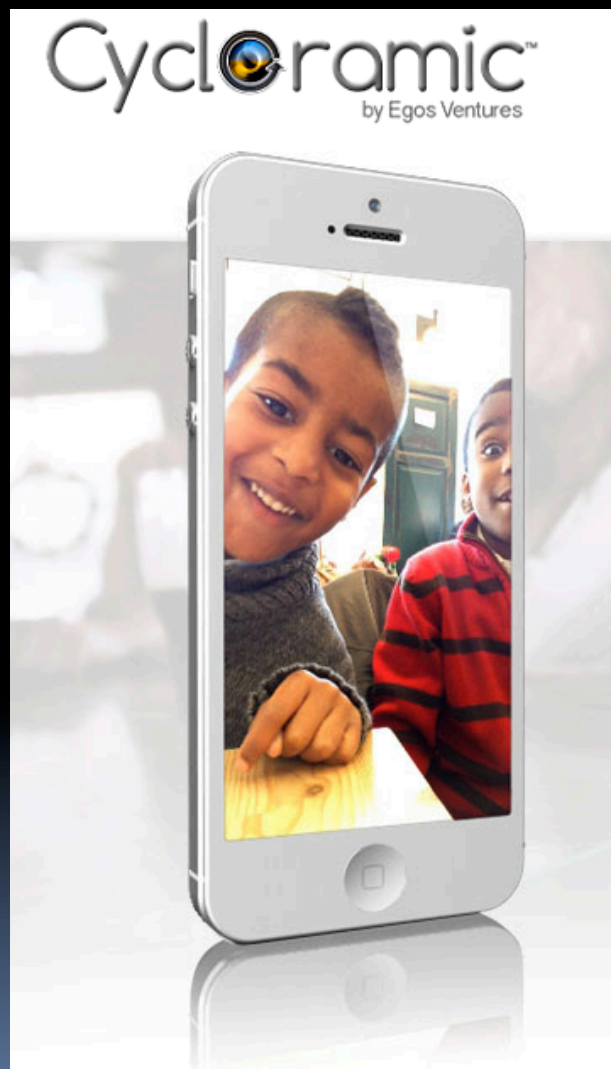
- ...of human expression and experience.
- **Computer music** can be created by synthesizing sounds, by sampling existing music, or by recording and manipulating sounds.
- **Creating digital effects, images, and animations** has impacted and **transformed the movie industry**.
- Computing enables creative **exploration of real and synthetic phenomena**.





Programs can be developed...

- ...for **creative expression** or to satisfy **personal curiosity**.
- A program developed for creative expression or to satisfy personal curiosity may have **visual, audible, or tactile results**; or the program may affect a computer or system without such results.
- Programs developed for creative expression or to satisfy personal curiosity may be developed with **different standards** or methods than programs developed for widespread distribution.
- A program or the results of running a program may be **shared with others**.





Programs can be developed...

- ...to **solve** problems, **create** new knowledge, or **help** people, organizations, or society.
 - however, the goals may be realized independently of the original purpose of the program.
- **Computer programs** and the results of running the programs have **widespread impact** on individuals, organizations, and society.





Numbers: Positional Notation

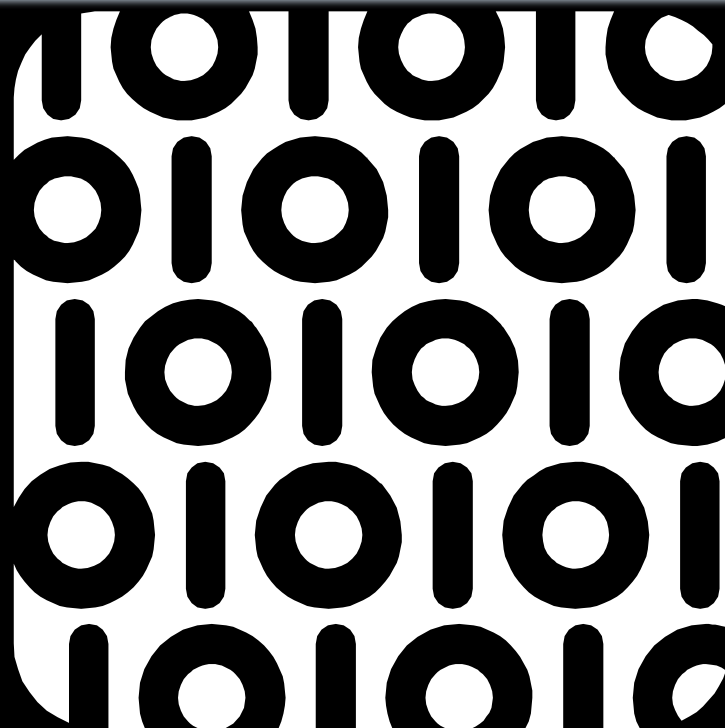
- Number Base $B \Rightarrow B$ symbols per digit:
 - Base 10 (Decimal): 0, 1, 2, 3, 4, 5, 6, 7, 8, 9
 - Base 2 (Binary): 0, 1 (In binary digits are called "bits")
 - Base 16 (Hexadecimal): 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F
- Number representation:
 - $d_{31}d_{30} \dots d_1d_0$ is a 32 digit number
 - value = $d_{31} \times B^{31} + d_{30} \times B^{30} + \dots + d_1 \times B^1 + d_0 \times B^0$
- Binary $0b11010 = 1 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0$
 $= 16 + 8 + 2$
 $= 26$
- Hex $0x1A = 1 \times 16^1 + 10 \times 16^0$
 $= 16 + 10$
 $= 26$
- One hex digit (four bits) is a "nibble". Two (eight bits) is a "byte" (values 0-255)
- N bits \Leftrightarrow at most 2^N things





Abstraction (revisited): Numbers

- **Number bases**, including binary and decimal, are used for reasoning about digital data.
- Bits represent binary data using **base two** digits: zero and one.
- **Hexadecimal**, or **base-16**, is often used in reasoning about data such as colors in images.
- **Different bases help** in reasoning about digital data; digital data is stored in bits.



Abstraction (revisited): Digital Data

- A **combination of abstractions** is used to represent digital data.
- At the lowest level **all digital data are represented by bits.**
 - Said another way, bits can represent anything!
- **Bits are grouped** to represent higher-level abstractions including numbers and characters.
 - Logical values? 0 → False, 1 → True
 - Colors? 00 → Red, 01 → Green, 10 → Blue
 - Characters? 00000 → 'a', 00001 → 'b', ...
- **Higher-level abstractions** such as Internet protocol (IP) packets, images, and audio files are comprised of groups of bits that represent different parts of the abstractions.



Binary Sequences to Represent Data

- A **finite representation** is used to model the infinite mathematical concept of a number.
- In many programming languages the **fixed number of bits** used to represent **integers** limits the range of integer values, and mathematical operations can result in **overflow** or other errors.
- In many programming languages the fixed number of bits used to represent **real numbers** (represented as “floating-point numbers”) **limits their range**, and mathematical operations can result in **round-off** and other errors.

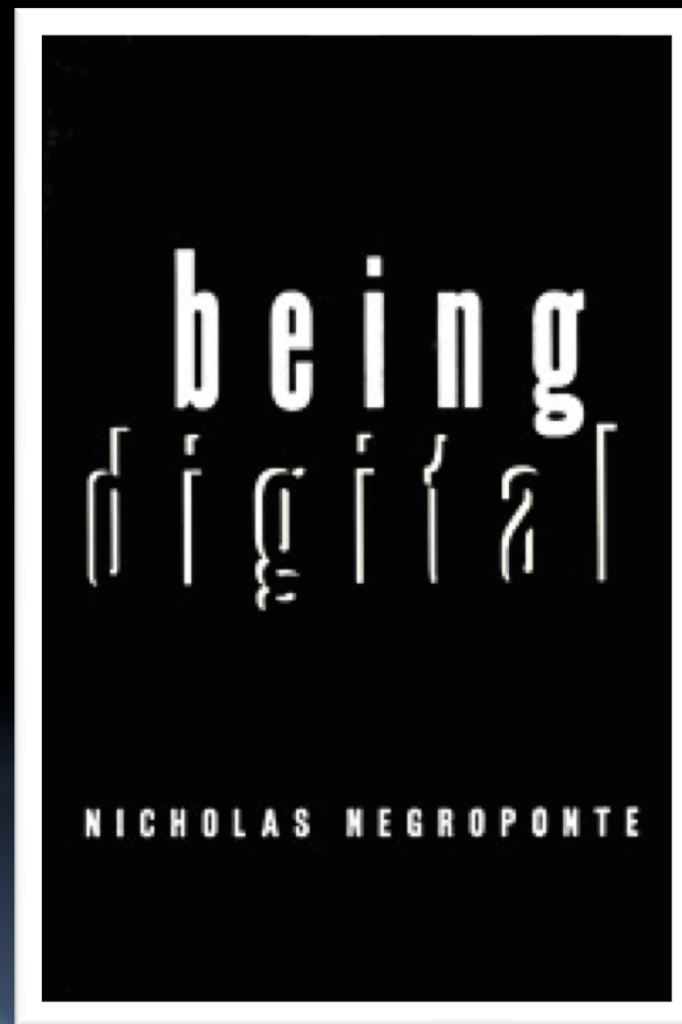
The screenshot shows the ACM Turing Award website for William ('Velvel') Morton Kahan. The page features the ACM logo, the text 'A.M. TURING AWARD', and a grid of award winners' portraits. Below the grid, there are navigation tabs: 'ALPHABETICAL LISTING', 'YEAR OF THE AWARD', and 'RESEARCH SUBJECT'. The main content area displays a portrait of Kahan, his name, and a citation: 'For his fundamental contributions to numerical analysis. One of the foremost experts on floating-point computations. Kahan has dedicated himself to "making the world safe for numerical computations"!'. There are also links for 'SHORT ANNOTATED BIBLIOGRAPHY', 'ACM DL AUTHOR PROFILE', 'RESEARCH SUBJECTS', and 'ADDITIONAL MATERIALS'. A detailed biography follows, mentioning his birth in 1933 in Toronto, Ontario, Canada, his education at the University of Toronto and the University of Waterloo, and his work at Chalmers Institute of Technology in Sweden. It also lists his awards, including the ACM Turing Award in 1973, and his role as a faculty member at the University of Toronto and the University of California, Berkeley.





Interpretation of a Binary Sequence...

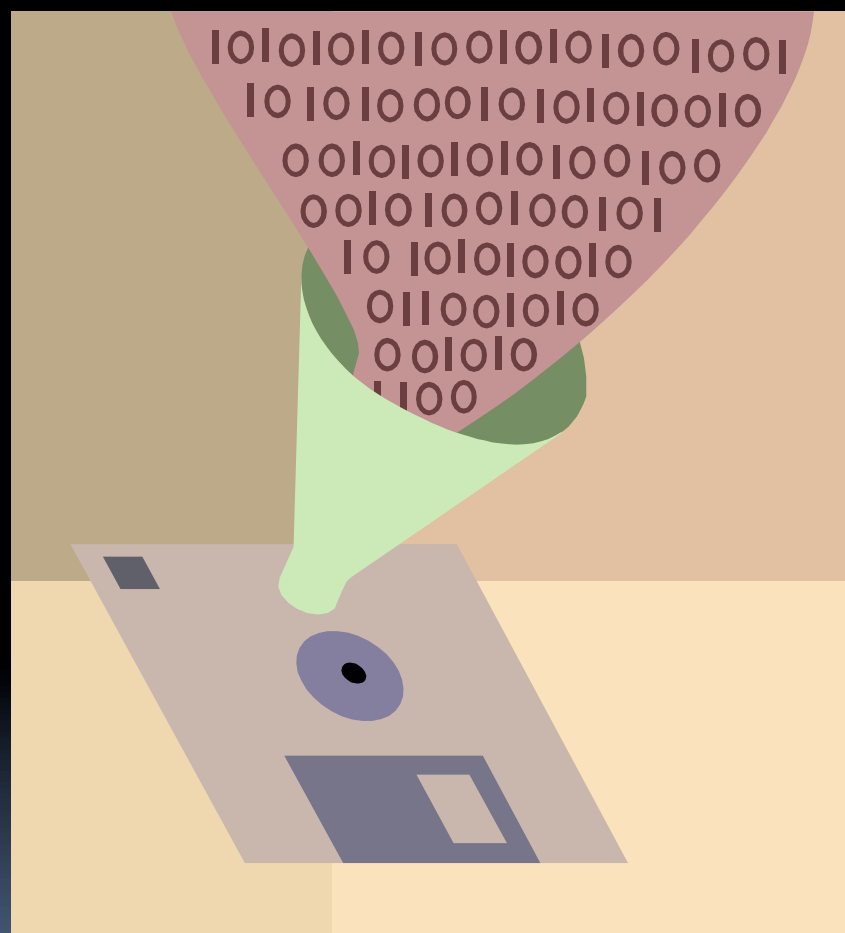
- ...**depends on how it is used** (e.g., as instruction, number, text, sound, or image).
- The sequence of bits that represents...
 - ...an instruction may also represent data processed by that instruction.
 - ...a character/letter may also represent a number.
 - ...a color in an image may also represent a sound in an audio file.







SW and HW built on multiple abstractions!

- **Software is built using low- and high-level abstractions...**
 - such as expressions, statements, data types, functions, and libraries.
 - that represent hardware, such as device drivers and game controllers.
- **Hardware is built using low- and high-level abstractions such as chips, memory, and storage.**





Binary Data is processed by...

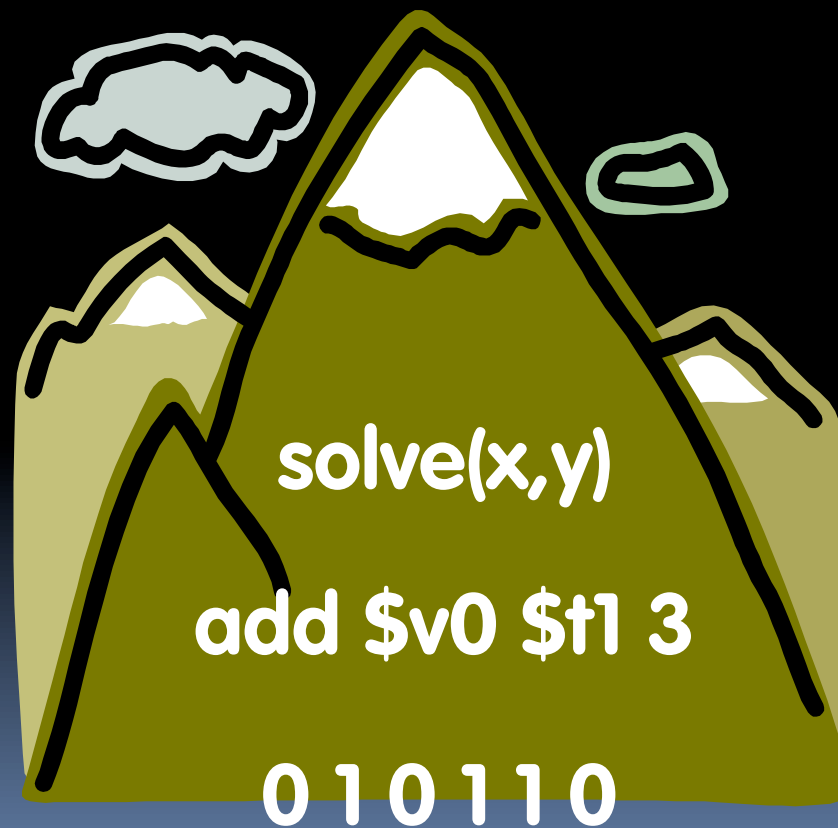
- ...physical layers of computing hardware, including gates, chips, and components.
- A **logic gate** is a hardware **abstraction** that models a Boolean function. 
- A **chip** is an **abstraction** composed of low-level components and circuits that performs a specific function such as memory, CPU, encryption, and more. 
- A hardware component can be low level like a transistor or high level like a video card.





Programming languages, from low to high level...

- ...are used in developing software.
- **Low-level programming languages**, such as assembly, are closer to the machine level and **provide fewer abstractions** for the programmer.
- **High-level programming languages** provide more abstractions for the programmer and **are easier for humans to use** for reading and writing code.
- Code in a high-level programming language is typically automatically translated into code in a lower-level language to be executed on a computer; this is done by a **compiler** or an **interpreter**.





Abstractions everywhere!

- Applications and systems are designed, developed, and analyzed using levels of hardware, software, and conceptual **abstractions**.
 - E.g., **Mobile applications** and systems
 - E.g., **Web services** (both an application and a system)





Summary

■ Creativity

- You will create interesting and relevant artifacts with the tools and techniques of computer science.



■ Abstraction

- This course will include examples of abstractions used in modeling the world, managing complexity, and communicating with people as well as with machines.
- You will learn to work with multiple levels of abstraction while engaging with computational problems and systems.

