Computer Science History

15-110 – Friday 04/18

Announcements

- **Check6-2** was due **today at noon**
  - Grades and feedback will be released on Sunday

- **Final Exam Logistics:**
  - Time: Monday 04/28 8:30-11:30am
  - Location: McConomy Auditorium
  - Same logistics as Exam1/Exam2, except you get **20 pages of notes** instead of 5

  - Study materials: https://www.cs.cmu.edu/~110/assessments.html
  - TA-Led review session: Sat 04/26 12pm in NSH 3305

Lecture Goal Overview

We're going to discuss four major revolutions that occurred in the past that led to the current computational era we live in.

1. Introduction of the theoretical concept of a computer
2. Construction of the first computer hardware and software
3. Transition of computers from government/corporate to personal
4. Connection of computers via the internet

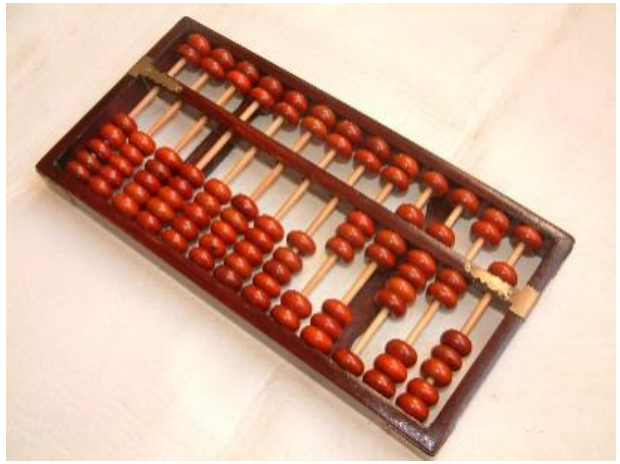
We won't ask you to memorize names or dates; instead, your goal is to recognize the most important components of each revolution. They will be listed as takeaways at the end of each section.

Generalized Computers and Algorithms

Computer Science and Math

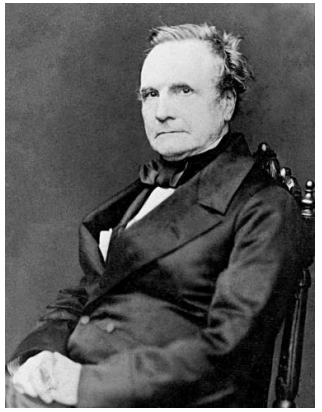
The foundation of the field of computer science lies in mathematics.

Many of the first 'computing devices' were analog machines built to do specific calculations on numbers, or automata made to accomplish tasks.



Pictured above: an abacus (2000 BC), Al-Jazari's castle clock automata (1206), and Hollerith's tabulating machine (1887)

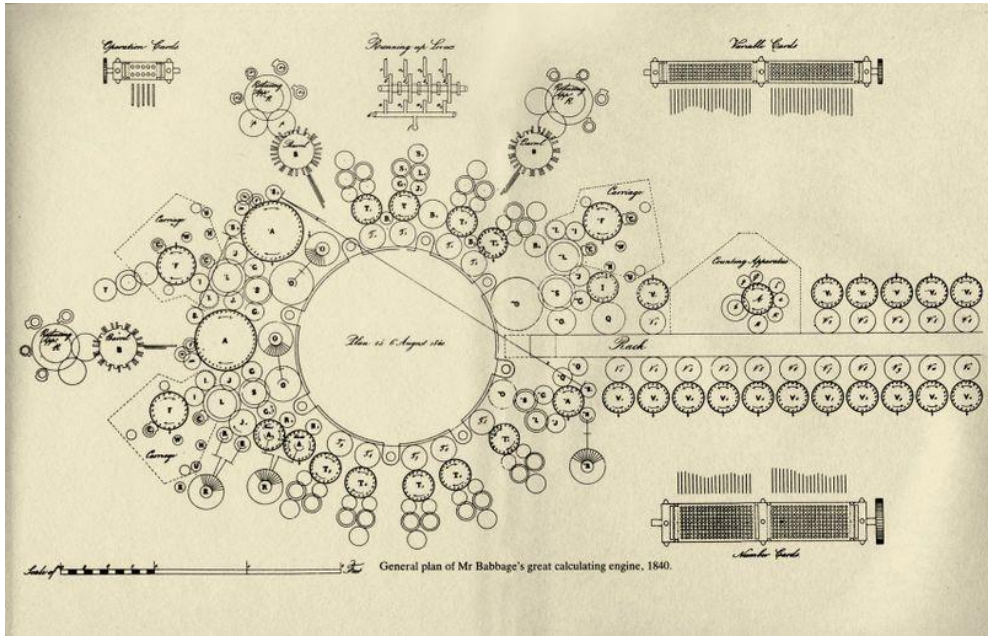
The Analytical Engine



In 1834-36, Charles Babbage designed the Analytical Engine.

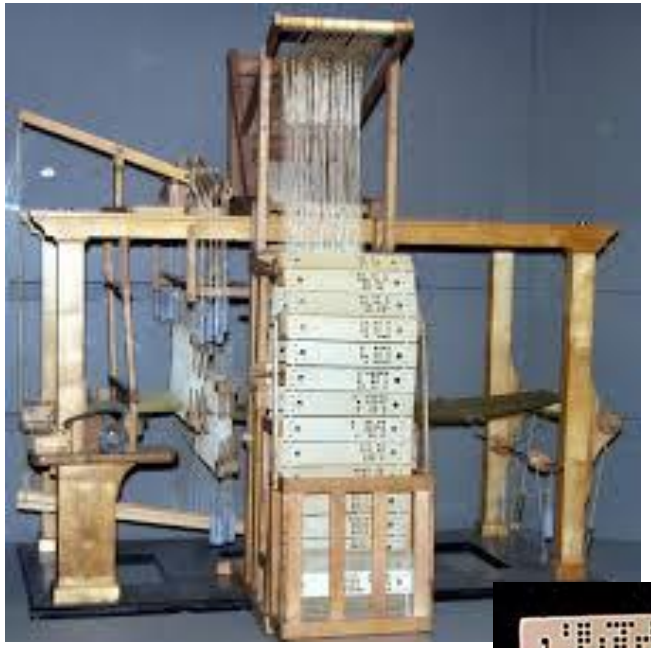
Many engineers and philosophers built computing machines and automata for specific purposes, but the Analytical Engine was the first design for a **general** computing device.

The goal: build a device capable of performing *any* mathematical calculation. Babbage's design incorporated features such as sequential statements, branches, and looping – all core parts of programming today!

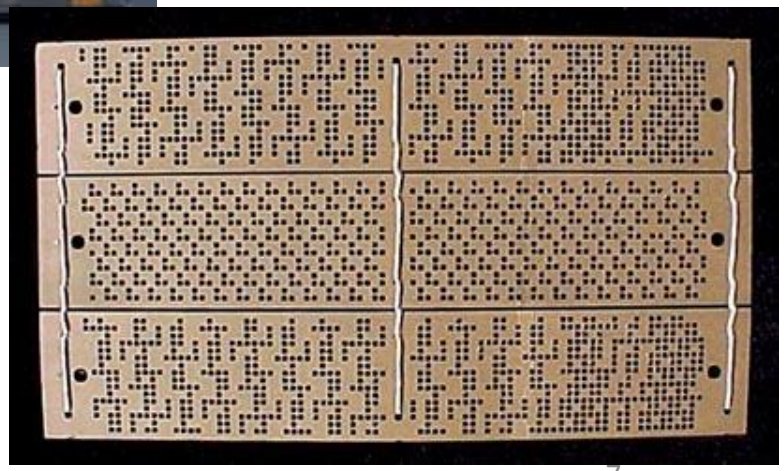


Punched Cards and Jacquard's Loom

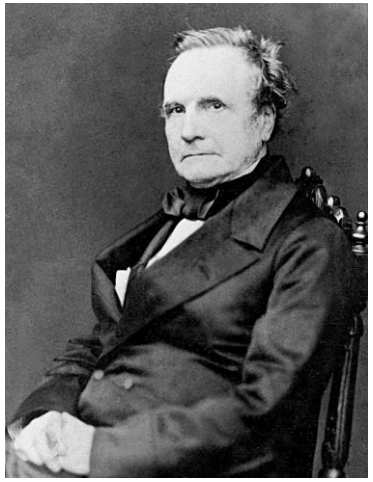
The Analytical Engine could be programmed using 'punched cards', a technology that had been developed to provide instructions for weaving on a mechanical loom in 1805 by Joseph-Marie Jacquard.



These cards could provide input for different weave patterns, to easily produce complex results. Babbage envisioned using them to provide instructions for a program.

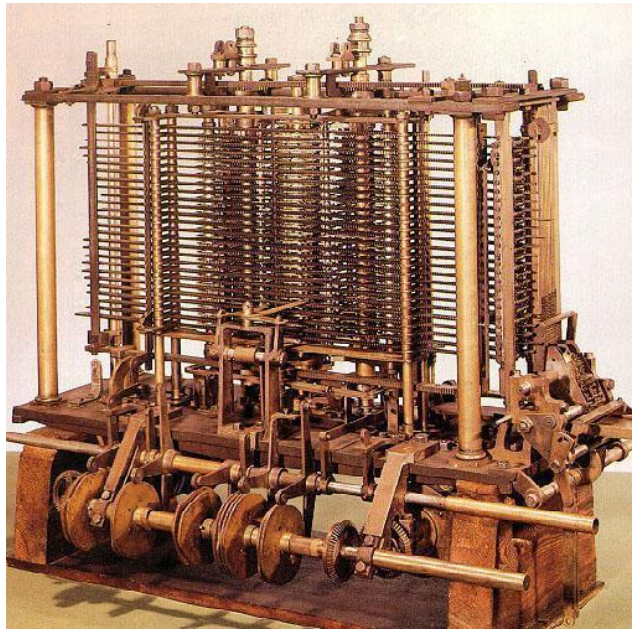


The Difference Engine



Unfortunately, Babbage was never able to build his Analytical Engine.

He did build an earlier machine, the Difference Engine, which could compute polynomial functions automatically. Looking at this device shows that the technology of the time was mechanical and hand-cranked, yet worked!



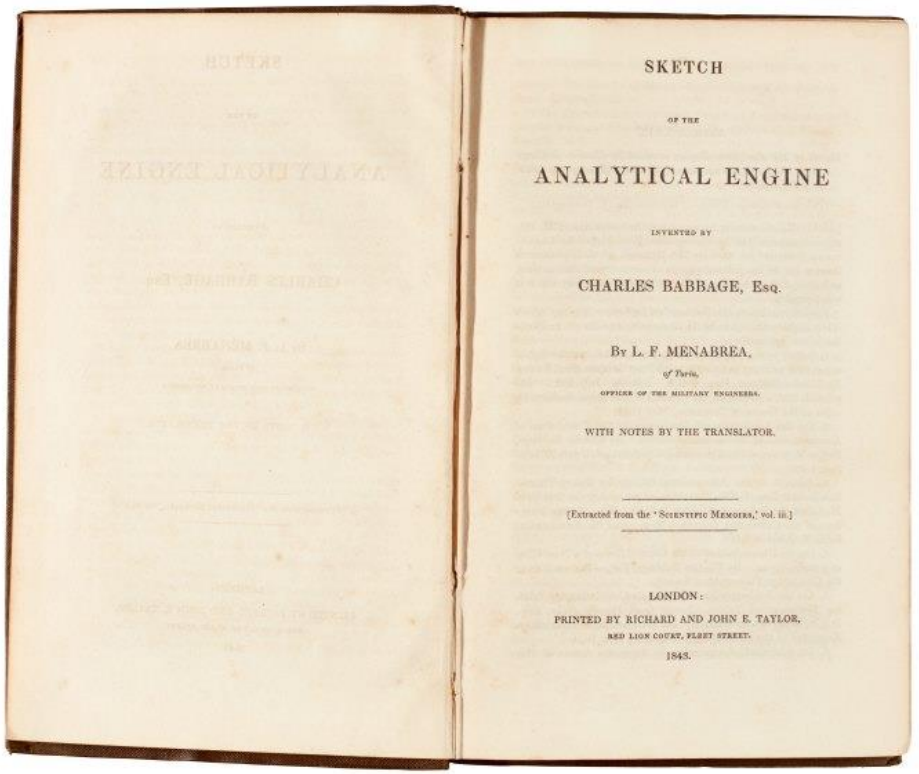
Here's a demo of a replica Difference Engine: https://www.youtube.com/watch?v=be1EM3gQkAY

The First Program

In 1843, one of Babbage's correspondent's, Ada Lovelace, was hired to translate lecture notes on the Analytical Engine from French to English.

She added extensive notes to this paper with her own thoughts. One of these notes contained an example that showed how the Analytical Engine could be used to calculate Bernoulli numbers.

This was the first program to be written for a computer, so Lovelace is considered the first programmer.



SKETCH

OF THE

ANALYTICAL ENGINE

INVENTED BY

CHARLES BABBAGE, Esq.

By L. F. MENABREA,

of Turin,

OFFICER OF THE MILITARY ENGINEERS.

WITH NOTES BY THE TRANSLATOR.

[Extracted from the 'Scientific Memoirs,' vol. iii.]

LONDON:

PRINTED BY RICHARD AND JOHN E. TAYLOR,

RED LION COURT, FLEET STREET.

1843.

Diagram for the computation by the Engine of the Numbers of Bernoulli. See Note G. (page 722 *et seq.*)

Number of Operation.	Nature of Operation.	Variables acted upon.	Variables receiving results.	Indication of change in the value on any Variable.	Statement of Results.	Data. ${}^1V_1$	Data. ${}^1V_2$	Data. ${}^1V_3$	Working Variables. ${}^0V_4$	${}^0V_5$	${}^0V_6$	${}^0V_7$	${}^0V_8$	${}^0V_9$	${}^0V_{10}$	${}^0V_{11}$	${}^0V_{12}$	${}^0V_{13}$	Result Variables. ${}^1V_{21}$ B_1 in a decimal fraction.	${}^1V_{22}$ B_2 in a decimal fraction.	${}^1V_{23}$ B_3 in a decimal fraction.	${}^0V_{24}$...
1	×	${}^1V_2 \times {}^1V_3$	${}^1V_4, {}^1V_5, {}^1V_6$	${}^1V_2 = {}^1V_2$, ${}^1V_3 = {}^1V_3$	$= 2n$	...	2	n	$2n$	$2n$	$2n$											
2	−	${}^1V_4 - {}^1V_1$	${}^2V_4$	${}^1V_4 = {}^2V_4$, ${}^1V_1 = {}^1V_1$	$= 2n-1$	1	...	...	$2n-1$													
3	+	${}^1V_5 + {}^1V_1$	${}^2V_5$	${}^1V_5 = {}^2V_5$, ${}^1V_1 = {}^1V_1$	$= 2n+1$	1	...	...	...	$2n+1$												
4	÷	${}^2V_5 \div {}^2V_4$	${}^1V_{11}$	${}^2V_5 = {}^0V_5$, ${}^2V_4 = {}^0V_4$	$= \frac{2n-1}{2n+1}$	...	...	...	0	0	...	...	...	...	...	$\frac{2n-1}{2n+1}$						
5	÷	${}^1V_{11} \div {}^1V_2$	${}^2V_{11}$	${}^1V_{11} = {}^2V_{11}$, ${}^1V_2 = {}^1V_2$	$= \frac{1}{2} \cdot \frac{2n-1}{2n+1}$	...	2	...	...	...	...	...	...	...	...	$\frac{1}{2} \cdot \frac{2n-1}{2n+1}$						
6	−	${}^0V_{13} - {}^2V_{11}$	${}^1V_{13}$	${}^2V_{11} = {}^0V_{11}$, ${}^0V_{13} = {}^1V_{13}$	$= -\frac{1}{2} \cdot \frac{2n-1}{2n+1} = A_0$	...	...	...	...	...	...	...	...	...	...	0	.....	$-\frac{1}{2} \cdot \frac{2n-1}{2n+1} = A_0$				
7	−	${}^1V_3 - {}^1V_1$	${}^1V_{10}$	${}^1V_3 = {}^1V_3$, ${}^1V_1 = {}^1V_1$	$= n-1\,(=3)$	1	...	n	...	...	...	...	...	...	$n-1$							
8	+	${}^1V_2 + {}^0V_7$	${}^1V_7$	${}^1V_2 = {}^1V_2$, ${}^0V_7 = {}^1V_7$	$= 2 + 0 = 2$	...	2	...	...	...	...	2										
9	÷	${}^1V_6 \div {}^1V_7$	${}^3V_{11}$	${}^1V_6 = {}^1V_6$, ${}^0V_{11} = {}^3V_{11}$	$= \frac{2n}{2} = A_1$	...	...	...	...	...	$2n$	2	...	...	...	$\frac{2n}{2} = A_1$						
10	×	${}^1V_{21} \times {}^3V_{11}$	${}^1V_{12}$	${}^1V_{21} = {}^1V_{21}$, ${}^3V_{11} = {}^3V_{11}$	$= B_1 \cdot \frac{2n}{2} = B_1 A_1$	...	...	...	...	...	...	...	...	...	...	$\frac{2n}{2} = A_1$	$B_1 \cdot \frac{2n}{2} = B_1 A_1$	.....	B_1			
11	+	${}^1V_{12} + {}^1V_{13}$	${}^2V_{13}$	${}^1V_{12} = {}^0V_{12}$, ${}^1V_{13} = {}^2V_{13}$	$= -\frac{1}{2} \cdot \frac{2n-1}{2n+1} + B_1 \cdot \frac{2n}{2}$ ...	...	...	...	...	...	...	...	...	...	...	.....	0	$\left\{-\frac{1}{2} \cdot \frac{2n-1}{2n+1} + B_1 \cdot \frac{2n}{2}\right\}$				
12	−	${}^1V_{10} - {}^1V_1$	${}^2V_{10}$	${}^1V_{10} = {}^2V_{10}$, ${}^1V_1 = {}^1V_1$	$= n-2\,(=2)$	1	...	...	...	...	...	...	...	...	$n-2$							
13	−	${}^1V_6 - {}^1V_1$	${}^2V_6$	${}^1V_6 = {}^2V_6$, ${}^1V_1 = {}^1V_1$	$= 2n-1$	1	...	...	...	...	$2n-1$											
14	+	${}^1V_1 + {}^1V_7$	${}^2V_7$	${}^1V_1 = {}^1V_1$, ${}^1V_7 = {}^2V_7$	$= 2 + 1 = 3$	1	...	...	...	...	...	3										
15	÷	${}^2V_6 \div {}^2V_7$	${}^1V_8$	${}^2V_6 = {}^2V_6$, ${}^2V_7 = {}^2V_7$	$= \frac{2n-1}{3}$	...	...	...	...	...	$2n-1$	3	$\frac{2n-1}{3}$									
16	×	${}^1V_8 \times {}^3V_{11}$	${}^4V_{11}$	${}^1V_8 = {}^0V_8$, ${}^3V_{11} = {}^4V_{11}$	$= \frac{2n}{2} \cdot \frac{2n-1}{3}$	...	...	...	...	...	...	...	0	...	...	$\frac{2n}{2} \cdot \frac{2n-1}{3}$						
17	−	${}^2V_6 - {}^1V_1$	${}^3V_6$	${}^2V_6 = {}^3V_6$, ${}^1V_1 = {}^1V_1$	$= 2n-2$	1	...	...	...	...	$2n-2$											
18	+	${}^1V_1 + {}^2V_7$	${}^3V_7$	${}^2V_7 = {}^3V_7$, ${}^1V_1 = {}^1V_1$	$= 3 + 1 = 4$	1	...	...	...	...	...	4										
19	÷	${}^3V_6 \div {}^3V_7$	${}^1V_9$	${}^3V_6 = {}^3V_6$, ${}^3V_7 = {}^3V_7$	$= \frac{2n-2}{4}$	...	...	...	...	...	$2n-2$	4	...	$\frac{2n-2}{4}$	...	$\left\{\frac{2n}{2} \cdot \frac{2n-1}{3} \cdot \frac{2n-2}{3}\right\} = A_2$							
20	×	${}^1V_9 \times {}^4V_{11}$	${}^5V_{11}$	${}^1V_9 = {}^0V_9$, ${}^4V_{11} = {}^5V_{11}$	$= \frac{2n}{2} \cdot \frac{2n-1}{3} \cdot \frac{2n-2}{4} = A_2$	...	...	...	...	...	...	...	...	0								
21	×	${}^1V_{22} \times {}^5V_{11}$	${}^0V_{12}$	${}^1V_{22} = {}^1V_{22}$, ${}^0V_{12} = {}^2V_{12}$	$= B_2 \cdot \frac{2n}{2} \cdot \frac{2n-1}{3} \cdot \frac{2n-2}{3} = B_2 A_2$	...	...	...	...	...	...	...	...	...	...	0	$B_2 A_2$	.....	.....	B_2		
22	+	${}^2V_{12} + {}^2V_{13}$	${}^3V_{13}$	${}^2V_{12} = {}^0V_{12}$, ${}^2V_{13} = {}^3V_{13}$	$= A_0 + B_1 A_1 + B_2 A_2$	...	...	...	...	...	...	...	...	...	...	.....	0	$\{A_2 + B_1 A_1 + B_2 A_2\}$				
23	−	${}^2V_{10} - {}^1V_1$	${}^3V_{10}$	${}^2V_{10} = {}^3V_{10}$, ${}^1V_1 = {}^1V_1$	$= n-3\,(=1)$	1	...	...	...	...	...	...	...	...	$n-3$							
Here follows a repetition of Operations thirteen to twenty-three.																						
24	+	${}^4V_{13} + {}^0V_{24}$	${}^1V_{24}$	${}^4V_{13} = {}^0V_{13}$, ${}^0V_{24} = {}^1V_{24}$	$= B_7$	...	...	...	...	...	...	...	...	...	...	.....	.....	.....	.....	.....	.....	B_7
25	+	${}^1V_1 + {}^1V_3$	${}^1V_3$	${}^1V_1 = {}^1V_1$, ${}^1V_3 = {}^1V_3$, ${}^5V_6 = {}^0V_6$, ${}^5V_7 = {}^0V_7$	$= n+1 = 4+1 = 5$ ..... by a Variable-card. by a Variable card.	1	...	$n+1$	...	...	0	0										

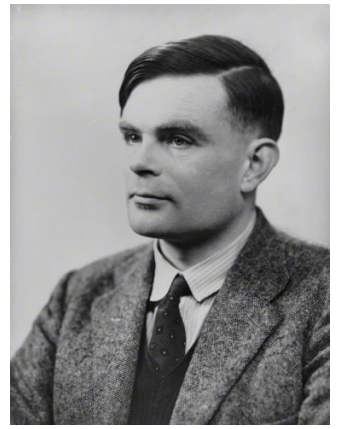
Computational Thinking

Ada Lovelace is also credited as being the first person to realize that computers could be used for more than just math. One of her notes read:

"[The Analytical Engine] **might act upon other things besides number**, were objects found whose mutual fundamental relations could be expressed by those of the abstract science of operations, and which should be also susceptible of adaptations to the action of the operating notation and mechanism of the engine...

Supposing, for instance, that the fundamental relations of **pitched sounds in the science of harmony and of musical composition** were susceptible of such expression and adaptations, **the engine might compose elaborate and scientific pieces of music** of any degree of complexity or extent."

A General Model of Computers



Actual physical computers would not be developed for another hundred years. But just before the first computers were built, in 1936, two people – Alonzo Church and Alan Turing – developed a general model of what can be computed (by today's computers). This is now referred to as the Church-Turing Thesis. We'll focus on Turing's model for now.

Turing invented the concept of a 'Turing Machine', a simple abstract machine that has wide-ranging capabilities. It is widely acknowledged today that all general-purpose computers can be reduced to the idea of a Turing Machine.

We also allow ourselves at any time to introduce abbreviations of the form that a particular symbol α shall stand for a particular sequence of symbols $\boldsymbol{A}$, and indicate the introduction of such an abbreviation by the notation $\alpha \rightarrow \boldsymbol{A}$, to be read, "$\alpha$ stands for $\boldsymbol{A}$."

We introduce at once the following infinite list of abbreviations,

$$1 \rightarrow \lambda ab \cdot a(b),$$
$$2 \rightarrow \lambda ab \cdot a(a(b)),$$
$$3 \rightarrow \lambda ab \cdot a(a(a(b))),$$

and so on, each positive integer in Arabic notation standing for a formula of the form $\lambda ab \cdot a(a(\cdot\ \cdot\ \cdot a(b) \cdot\ \cdot\ \cdot))$.

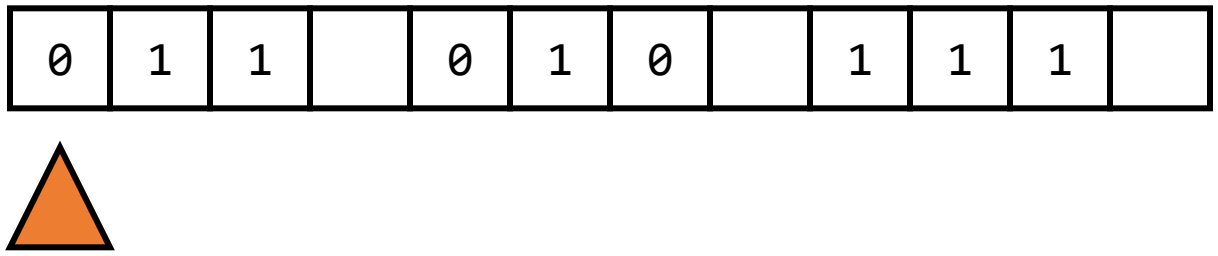
6. *The universal computing machine.*

It is possible to invent a single machine which can be used to compute any computable sequence. If this machine $\mathfrak{U}$ is supplied with a tape on the beginning of which is written the S.D of some computing machine $\mathcal{M}$,

SER. 2. VOL. 42. NO. 2144. R

Turing Machines

A Turing machine can be thought of as a long piece of tape combined with a device.

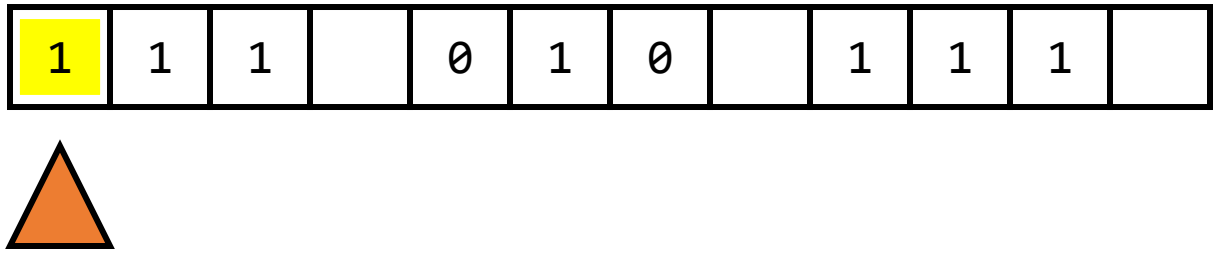


The tape is divided into cells. Each cell can either be blank or can have a symbol written in it.

The device can move to any cell on the tape. It can read the current symbol in the cell, erase the current value, or write a new value.

Turing Machines

The Turing Machine has a set of **states** used to determine what to do next.



Based on its current state and the value in the cell the device points to, it can change state, change the value in the cell, and/or move the device.

This perfectly models our computers today!

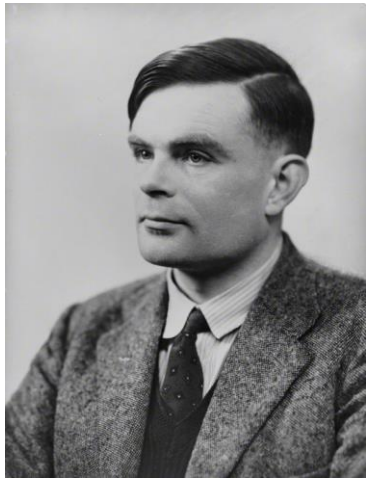
```
State A:
    If 0:
        Set to 1
        Move right one position

    If 1:
        Go to State D

    If blank:
        End process

State B:
        ...
```

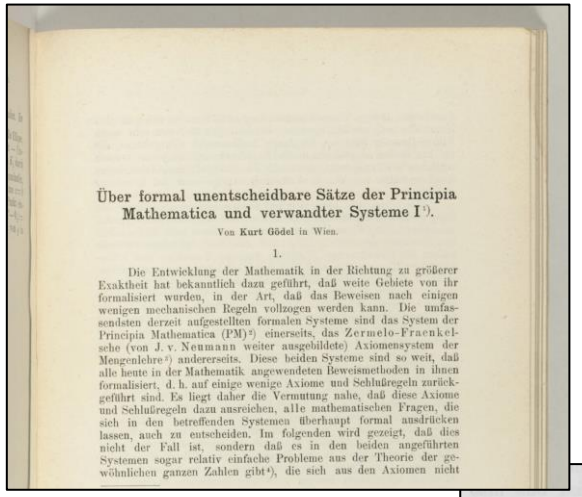
Limits of Computation



At the same time, Alan Turing proved that a Turing Machine (computer) can never solve certain problems.

Earlier (in 1930), Kurt Gödel proved the Incompleteness Theorem, which showed that every formal system will have some expressions it cannot represent.

Turing demonstrated that not everything is computable by formulating the Halting problem. He used a proof by contradiction to show that it's impossible to write a program that can **always** determine whether another program with a given input will ever halt (stop).

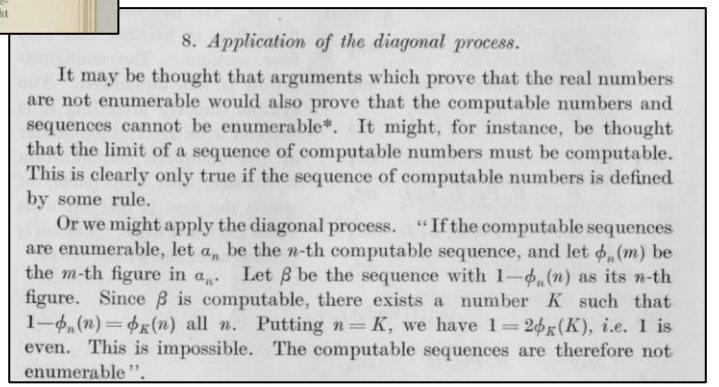


Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme I[1].

Von Kurt Gödel in Wien.

1.

Die Entwicklung der Mathematik in der Richtung zu größerer Exaktheit hat bekanntlich dazu geführt, daß weite Gebiete von ihr formalisiert wurden, in der Art, daß das Beweisen nach einigen wenigen mechanischen Regeln vollzogen werden kann. Die umfassendsten derzeit aufgestellten formalen Systeme sind das System der Principia Mathematica (PM)[2] einerseits, das Zermelo-Fraenkelsche (von J. v. Neumann weiter ausgebildete) Axiomensystem der Mengenlehre[3] andererseits. Diese beiden Systeme sind so weit, daß alle heute in der Mathematik angewendeten Beweismethoden in ihnen formalisiert, d. h. auf einige wenige Axiome und Schlußregeln zurückgeführt sind. Es liegt daher die Vermutung nahe, daß diese Axiome und Schlußregeln dazu ausreichen, alle mathematischen Fragen, die sich in den betreffenden Systemen überhaupt formal ausdrücken lassen, auch zu entscheiden. Im folgenden wird gezeigt, daß dies nicht der Fall ist, sondern daß es in den beiden angeführten Systemen sogar relativ einfache Probleme aus der Theorie der gewöhnlichen ganzen Zahlen gibt[4]), die sich aus den Axiomen nicht



8. *Application of the diagonal process.*

It may be thought that arguments which prove that the real numbers are not enumerable would also prove that the computable numbers and sequences cannot be enumerable*. It might, for instance, be thought that the limit of a sequence of computable numbers must be computable. This is clearly only true if the sequence of computable numbers is defined by some rule.

Or we might apply the diagonal process. "If the computable sequences are enumerable, let α_n be the n-th computable sequence, and let $\phi_n(m)$ be the m-th figure in α_n. Let β be the sequence with $1-\phi_n(n)$ as its n-th figure. Since β is computable, there exists a number K such that $1-\phi_n(n)=\phi_K(n)$ all n. Putting $n=K$, we have $1=2\phi_K(K)$, *i.e.* 1 is even. This is impossible. The computable sequences are therefore not enumerable".

Main Takeaways

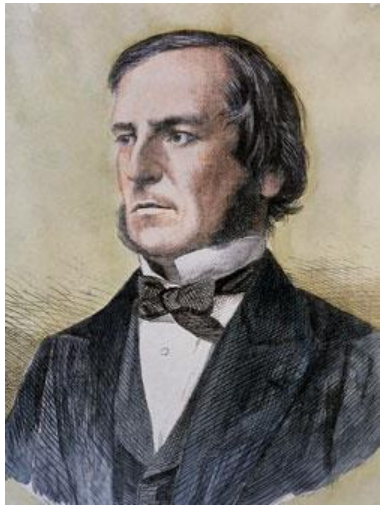
- The first design for a general computing device was the Analytical Engine. It was invented in the 19th century.

- The first program was written shortly afterwards. It aimed to calculate Bernoulli numbers on the Analytical Engine.

- The general definition of what can be computed can be stated using the Church-Turing Thesis through the concept of a Turing Machine, which can compute anything a digital computer can compute.

Original Hardware and Software

Boolean Algebra



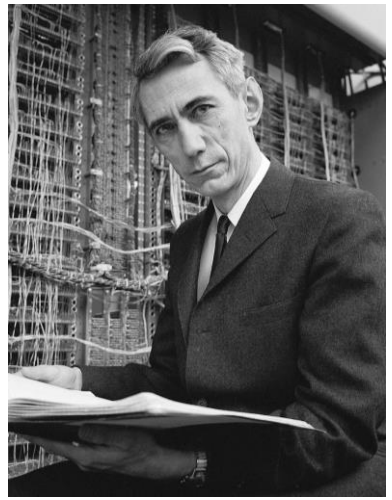
In 1854, George Boole published "An Investigation of the Laws of Thought", which first introduced the idea of Boolean algebra and logic.

He recognized the useful properties of addition and multiplication on 0s and 1s, as well as the 'and' and 'or' operations.

Boolean values are named after him!

6. Speaking generally, the symbol + is the equivalent of the conjunctions "and," "or," and the symbol −, the equivalent of the preposition "except." Of the conjunctions "and" and "or," the former is usually employed when the collection to be described forms the subject, the latter when it forms the predicate, of a proposition. "The scholar *and* the man of the world desire happiness," may be taken as an illustration of one of these cases. "Things possessing utility are *either* productive of pleasure *or* preventive of pain," may exemplify the other. Now whenever an expression involving these particles presents itself in a primary proposition, it becomes very important to know whether the groups or

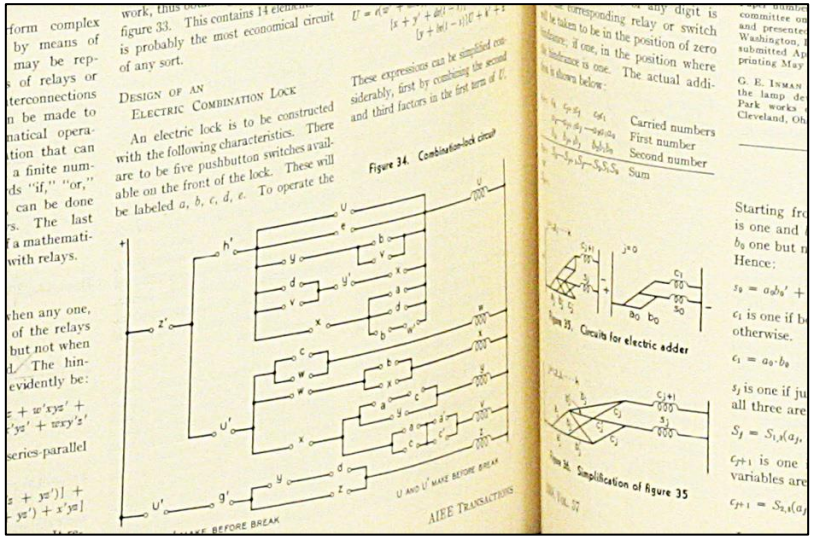
From Booleans to Circuits



In 1937, Claude Shannon published his Master's Thesis, "A Symbolic Analysis of Relay and Switching Circuits". This was the first time Boolean logic was translated into a physical format with electronics.

This work became the foundation of circuit design and made it possible to design the computers we know today.

Shannon also invented the full adder as an example in his paper!



Using the method of symmetric functions, and shifting down for s_j gives the circuits of Figure 35. Eliminating superfluous elements we arrive at Figure 36.

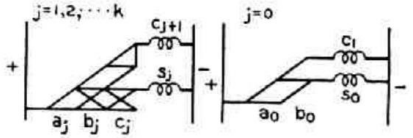


Figure 35. Circuits for electric adder

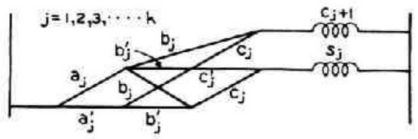


Figure 36. Simplification of figure 35

Computing Devices in World War II

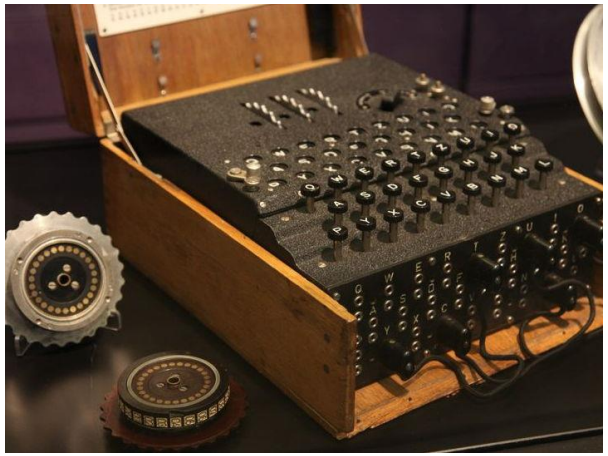
Shortly after this electronic breakthrough, World War II began. Computing was used to gain advantages in wartime efforts.

Computing played the most powerful role in code-breaking, as Allied forces attempted to decipher German messages (and vice versa).

We'll take a brief dive into work done in Great Britain, at Bletchley Park, which led to the first programmable computer.

Encryption and War Efforts

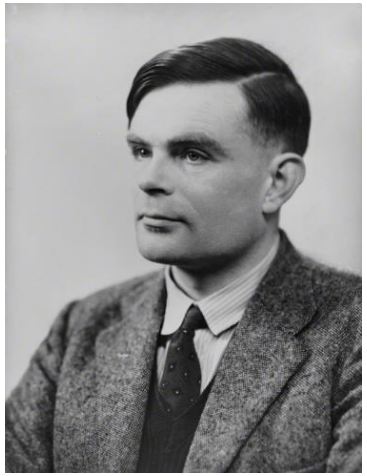
The German forces used a device called the Enigma Machine to encrypt communications. This encryption used a type of substitution cipher with a **shared key** (symmetric encryption!). German officers were given key lists ahead of time and would set a new key every day.



The Allied forces were able to reconstruct the physical device. However, they had to check all possible keys by hand every day, which took too long to be useful.

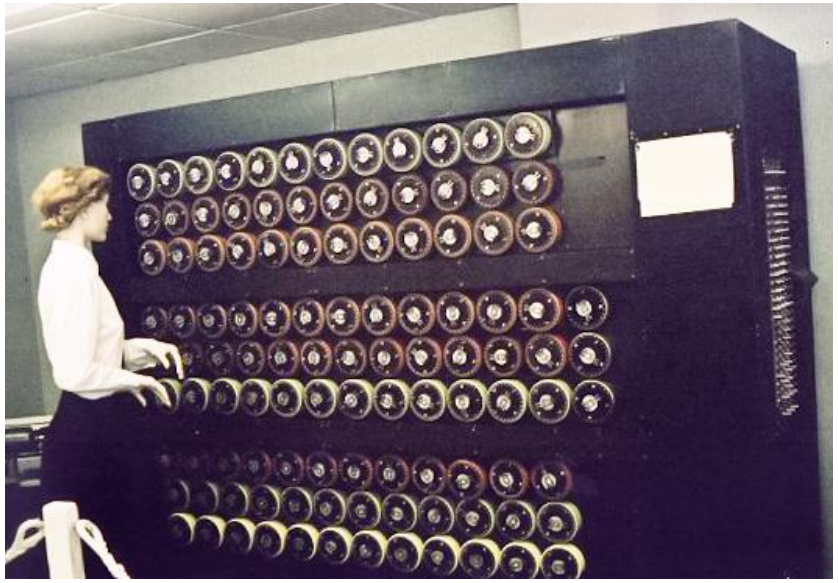
This lasted until someone noticed a pattern in German messages – they always sent a weather report at 6am each day. The common words in this report made it easier to check possible keys **computationally**.

The Bombe



The original deciphering machine, the Bomba, was designed by Marian Rejewski in Poland in 1938. Due to improvements in the Enigma and a lack of funds, the idea was passed to Britain.

In 1939, Alan Turing worked with a team to develop the Bombe, which checked all possible settings to see if they could find one that matched the expected words.



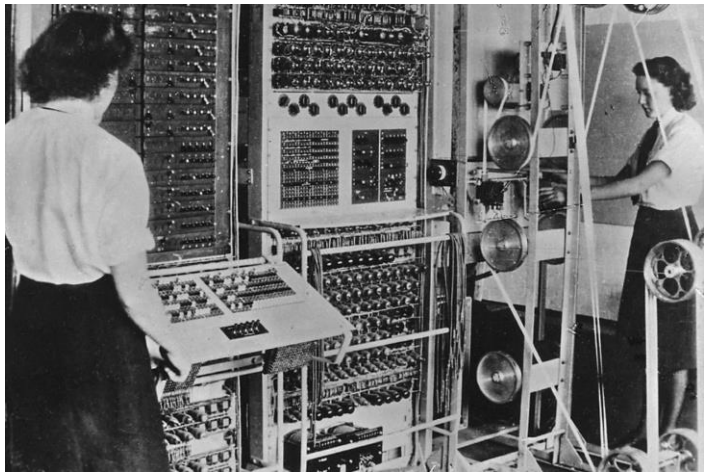
This process was dramatized in the movie The Imitation Game.

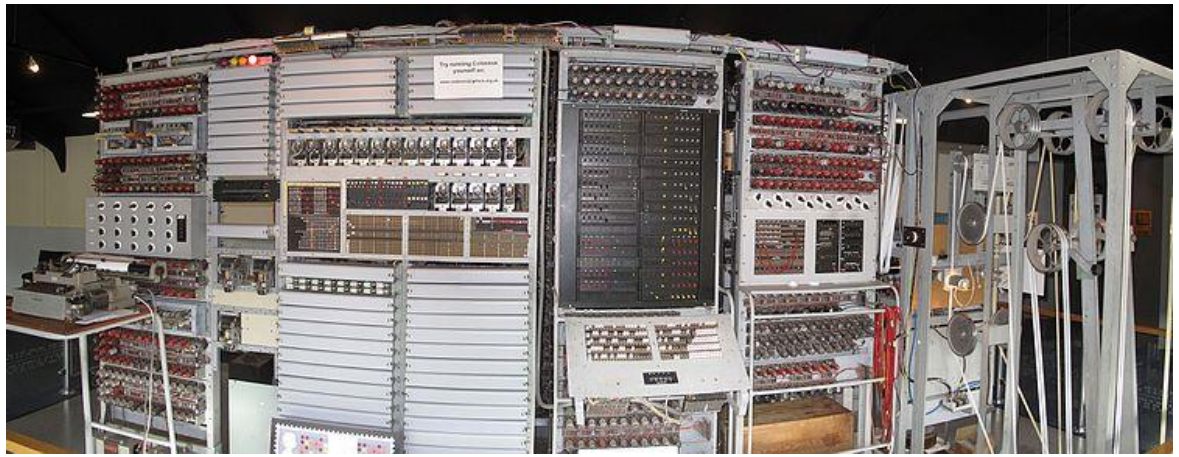
- https://www.youtube.com/watch?v=eYfCvBDVSQY

The Colossus

Later in the war, German forces started using a new encryption system for high-security messages. The Lorenz cipher proved much harder to crack, as the Allied forces had no information about the machine used to produce them.

From 1943-1945, Tommy Flowers led a team to design the Colossus, which was used to break Lorenz ciphers. This is widely considered to be the first **electronic programmable** computer. However, it could only be programmed for cipher-breaking, not general tasks.



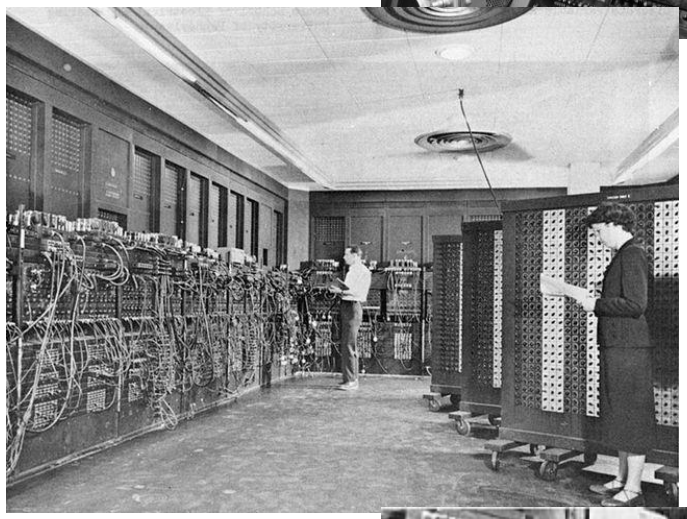


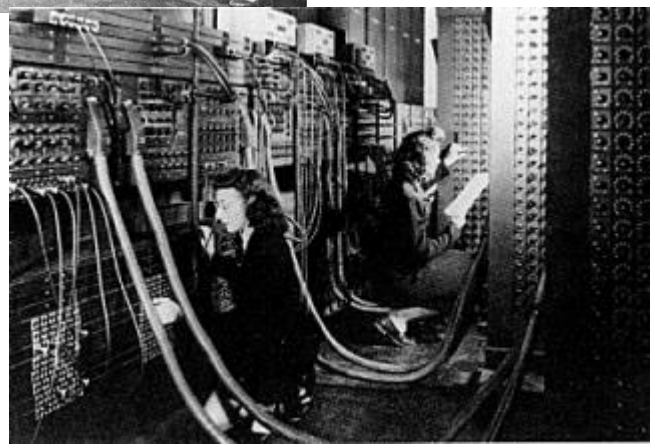
The First Modern Computer

In 1945, after the war ended, companies and research groups started work on designing computers for corporate and military use.

John Mauchly and J. Presper Eckert designed the ENIAC (Electronic Numerical Integrator and Computer), one of the first electronic **general-purpose** computers. It influenced many machines that came after it.

This machine was programmable (by moving wires) and had input and output in the form of punch cards. It could only hold up to 200 decimal digits in memory at first. That's around 80 bytes!

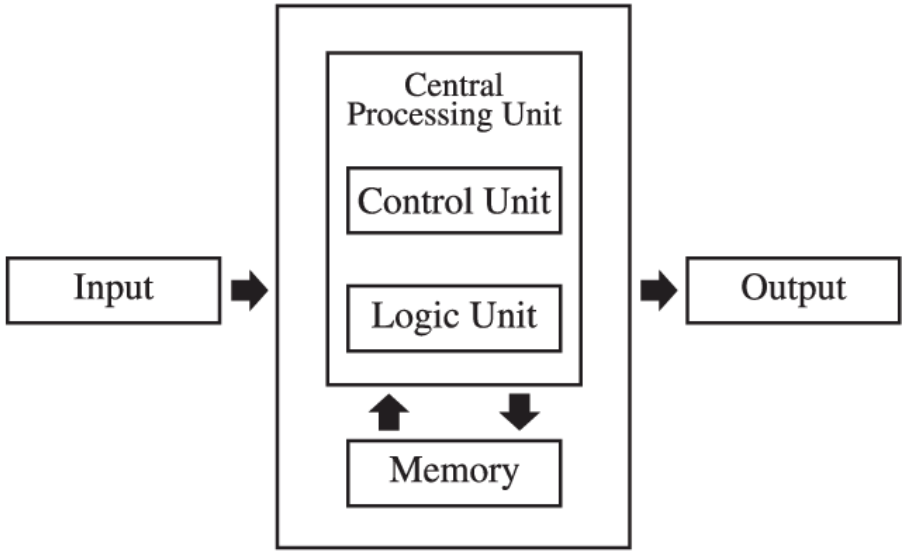




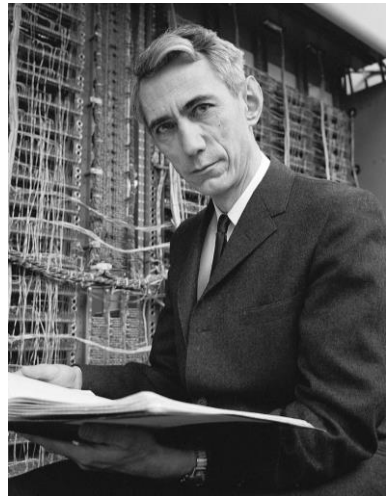
Modern Software Architecture

With the introduction of general-purpose computers came the need for software systems to support programming. Still in 1945, the software architecture of computers that we use today was designed.

John von Neumann introduced the von Neumann architecture, which organized the CPU, memory, and input/output. This also introduced the idea of representing machine code by running instructions sequentially until a conditional jump is reached.



Information Theory

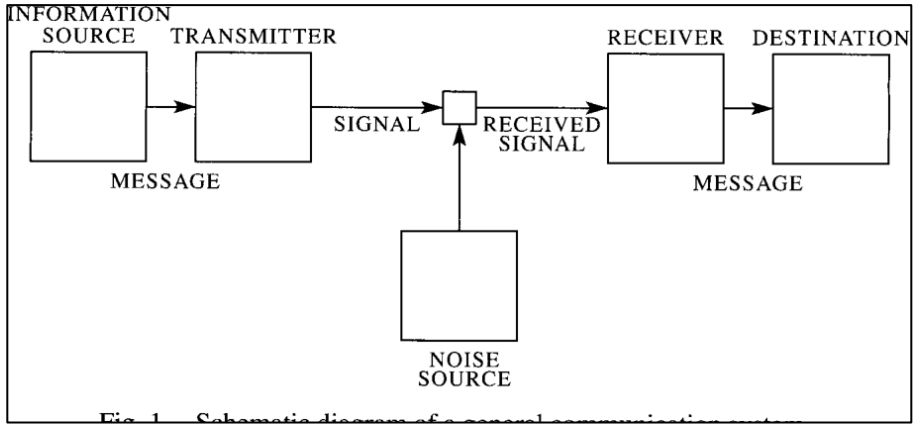


In 1948, Claude Shannon used mathematics to model core ideas in computing.

He published "A Mathematical Theory of Communication", which introduced many of the core ideas of **abstraction** and encoding we use today.

Introduced concepts of encoding, compression, and the bit!

Shannon is considered the father of information theory.

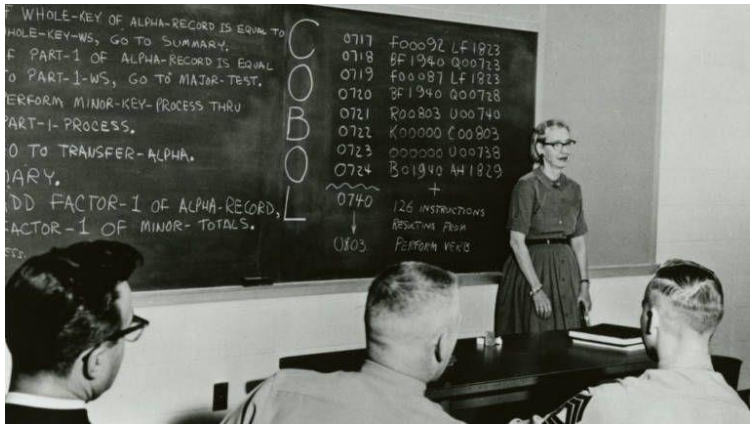


Programming Languages

Up until this point, machines ran on punch-card or typed code that was machine-specific and not very readable.

In 1952, Grace Hopper invented a compiler that could take statements written in strictly formatted English and translate them into computer-readable code. This made it much easier to write new programs for computers.

She also was part of a team that designed COBOL, one of the first plain-language programming languages.



Main Takeaways

- The invention of electronic circuitry made it possible to build physical computers
- WWII led to the design and implementation of programmable computers
- The ENIAC was the first general programmable computer
- The von Neumann architecture was a breakthrough software architecture that we still use today
- The compiler led to programs becoming easier to write

Personal Computing

Corporate to Personal

Originally, computers were only used for corporate or government purposes. Individuals did not own computers, because they were far too large and difficult to interact with.

This changed due to two events: invention of technology that made computers smaller, and invention of interaction modalities that made computers easier to work with.

The Transistor

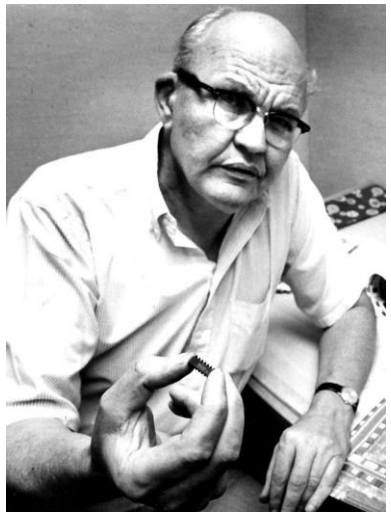
In 1947, John Bardeen, William Shockley and Walter Brattain at AT&T Bell Labs designed the transistor. This device could be used to switch electric signals.

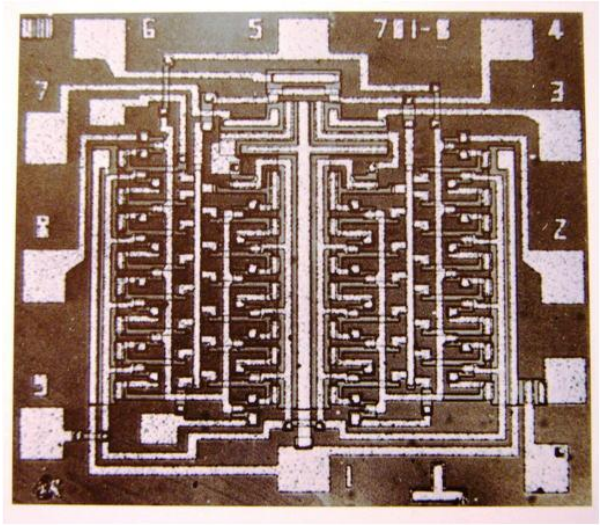
Previously, computers had to use vacuum tubes, which were very large. The invention of the transistor made it possible to make computers smaller.

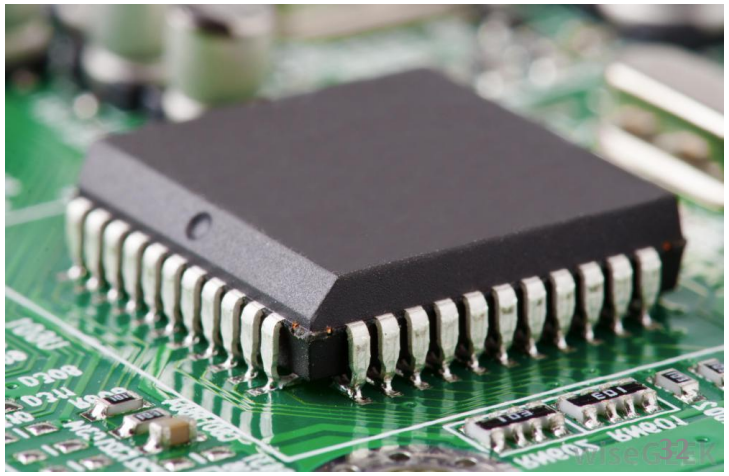
The Integrated Circuit

In 1958, Jack Kilby invented the Integrated Circuit (IC). This is a small electronic device (or 'chip') that can contain a large number of circuits and is easy to produce. It was possible to make ICs because of the invention of the transistor.

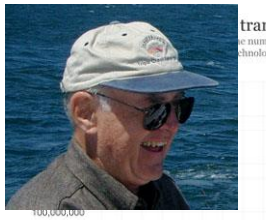
The IC again made it possible to make computers much smaller, as more electronics could be fit onto a smaller surface.







Moore's Law and The Microprocessor



In 1965, Gordon Moore introduced the business model known as Moore's Law, which states that the number of transistors on an IC will double every two years.

By 1971, this led to the invention of the microprocessor at Intel. A microprocessor is a whole processor that can fit onto a single chip.

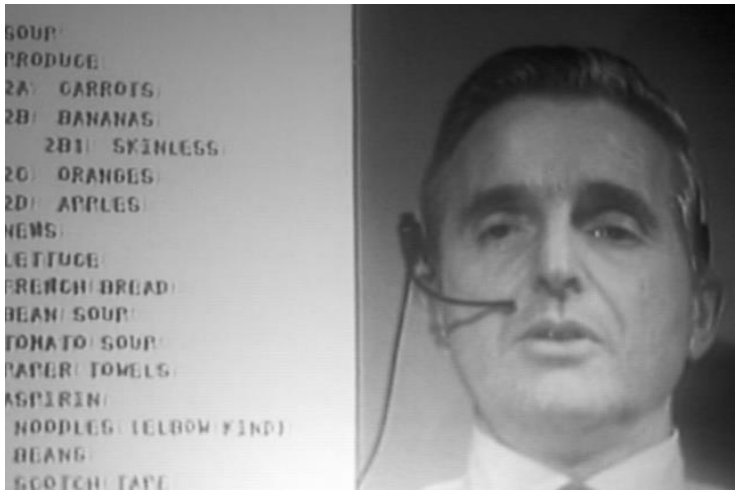
This breakthrough made it possible to put chips in many new devices, like calculators and clocks!

The Mother of All Demos

In 1968, Douglas Engelbart presented work he had done at the Augmentation Research Center at Stanford to a group of engineers at a computer conference. This 1.5 hour presentation later became known as the Mother of All Demos because it introduced an astounding number of technologies that we use to this day.

You can watch the demo for yourself online:

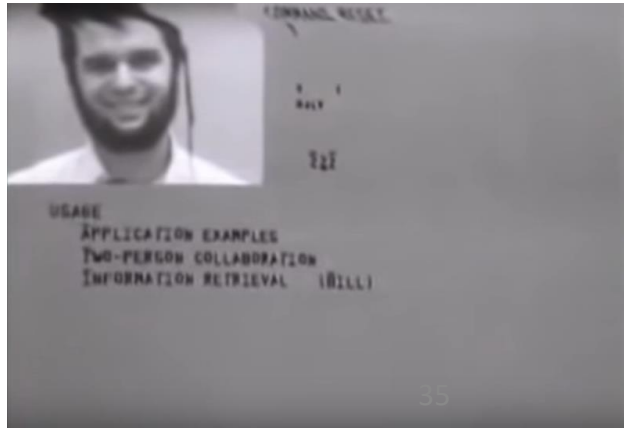
https://www.youtube.com/watch?v=yJDv-zdhzMY



Engelbart's Inventions

The technologies Engelbart introduced in this live demo include:

- The computer mouse
- The GUI (Graphical User Interface)
- The WYSIWIG (What You See Is What You Get) text editor
- The concept of multiple windows
- Revision control
- Video conferencing
- Real-time collaborative editing

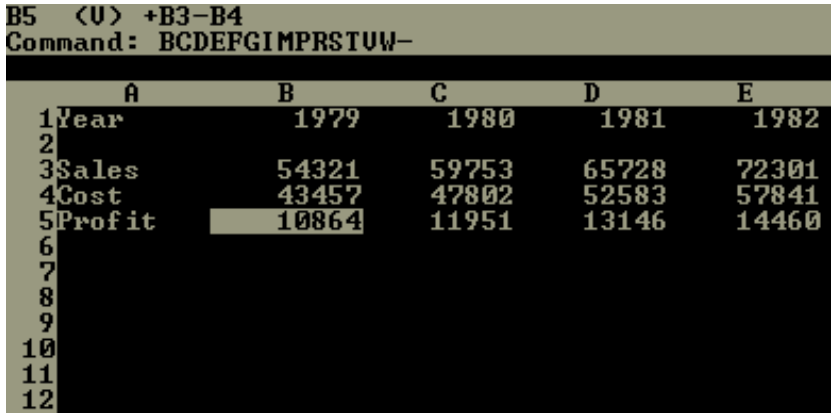


The First Killer App

In 1979, the application VisiCalc was produced by VisiCorp. This was a basic spreadsheet application that let the user modify values in a table and automatically re-calculate the results.

This was a huge development for business, as re-calculating tables by hand took teams of accountants long periods of time. The application became widely popular.

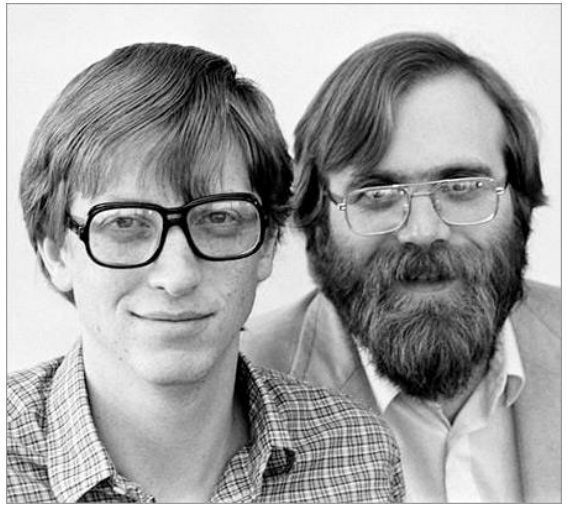
The launch of VisiCalc led to a boom in the personal computing business, and further competition.



Computing Companies

In 1975, Bill Gates and Paul Allen founded Microsoft. The company originally provided software (an interpreter for the language BASIC) to IBM.

Microsoft wanted to get into the personal computing business after seeing Apple's success.



In 1976, Steve Jobs and Steve Wozniak founded Apple. The company originally built and programmed the Apple I, one of the first personal computing devices, which was originally entirely text-based.

GUI Drama

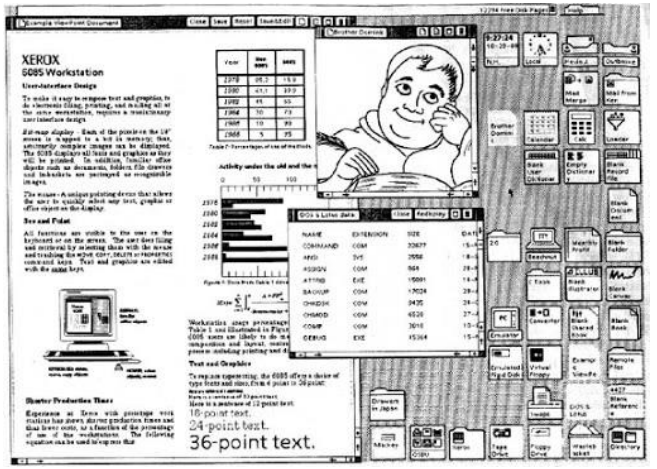
After the Mother of All Demos in 1968, several people on Engelbart's team went to work at Xerox PARC, to further develop the concepts. But Xerox didn't know what to do with their work.

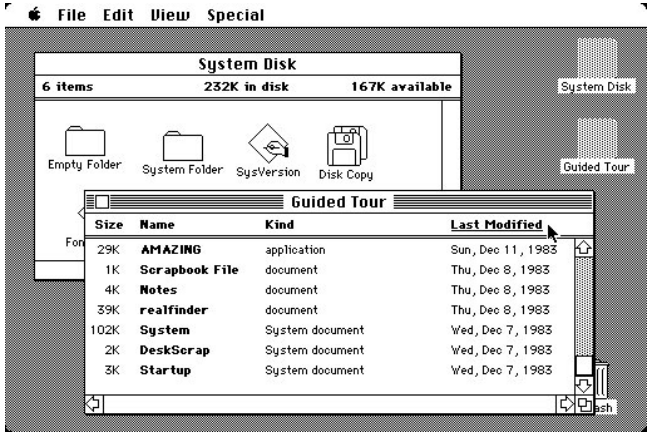
In 1979, Apple employees visited PARC and were shown their GUI implementation. They implemented similar ideas in the Apple Macintosh (released in 1984) to great acclaim.

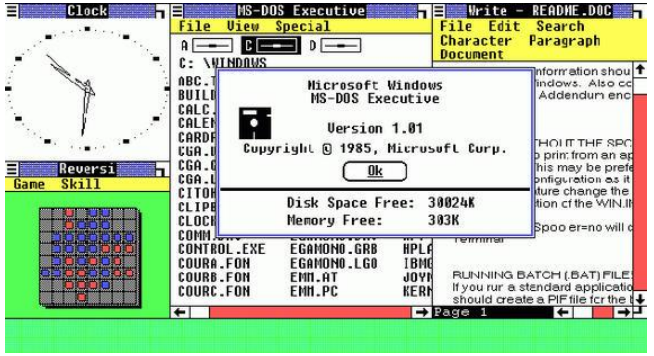
In 1981, Microsoft visited Apple and helped them develop some apps. They took the GUI idea from Apple and used it in their first operating system, MS-DOS, released in 1985.

This was dramatized in the 1999 film "Pirates of Silicon Valley":

https://www.youtube.com/watch?v=CBri-xgYvHQ







Main Takeaways

- The invention of the transistor, the integrated circuit, and the microprocessor made it possible to make computers smaller and cheaper.

- Many of the core ideas used in personal computing came from the Mother of All Demos and were later adopted by major computing companies.

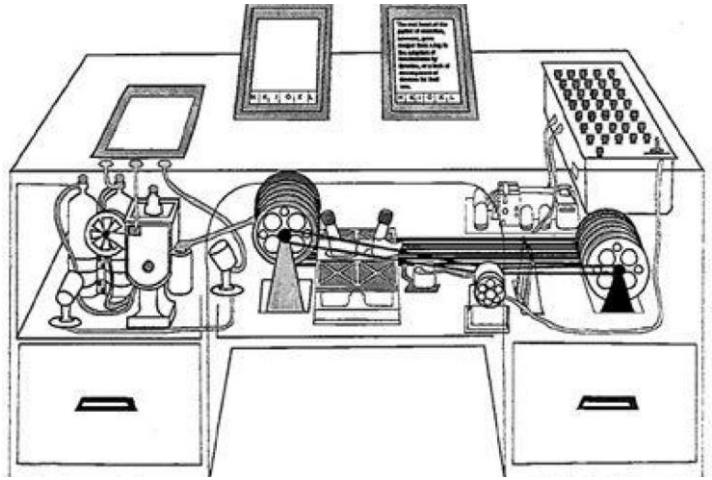
The Internet

Ideating the Internet – the Memex

Some of the core concepts of how the internet would work were introduced well before it was implemented.

In 1945, Vannevar Bush published As We May Think, which envisioned a system (Memex) to aid in research work. He invented the concept of hypertext!

"Consider a future device... in which an individual stores all his books, records, and communications, and which is mechanized so that it may be consulted with exceeding speed and flexibility. It is an enlarged intimate supplement to his memory."



ARPANET

In 1969, the US military wanted to create a decentralized communication system so that communications could not be knocked out entirely by a nuclear attack.

DARPA (Defense Advanced Research Projects Agency) collaborated with several universities to build the ARPANET, the Advanced Research Projects Agency Network. The initial network only connected four universities, but it grew over time.

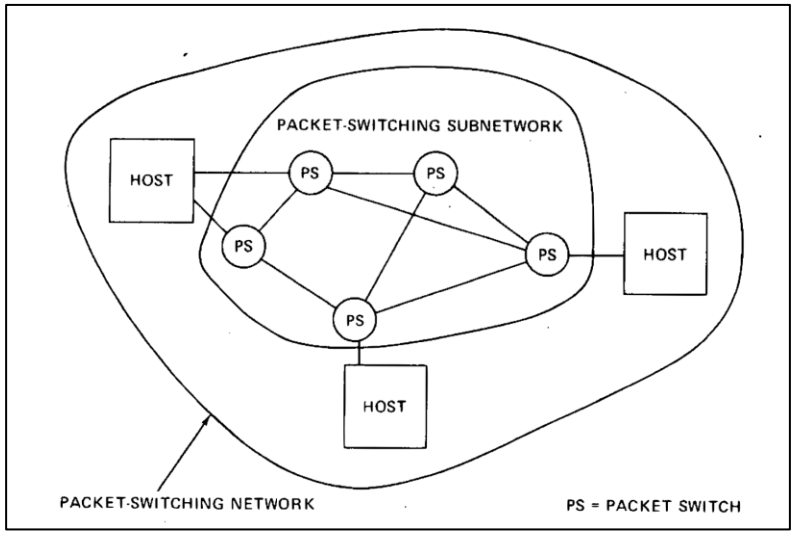
This system first introduced the concept of packets.

Communication Protocols

In 1982, Vinton Cerf and Robert Kahn designed and advocated for the TCP/IP protocol. TCP organizes data that is being sent between computers; IP delivers that data to the correct destination (based on IP addresses!).

The invention of TCP/IP made it much easier to connect computers together, which helped ARPANET expand its reach. Because of this, Vint Cerf and Bob Kahn are known as the fathers of the internet.

By 1984, the US military broke off from ARPANET to form their own private network (MILNET). More organizations and companies started to join the public network, forming the internet as we know it.



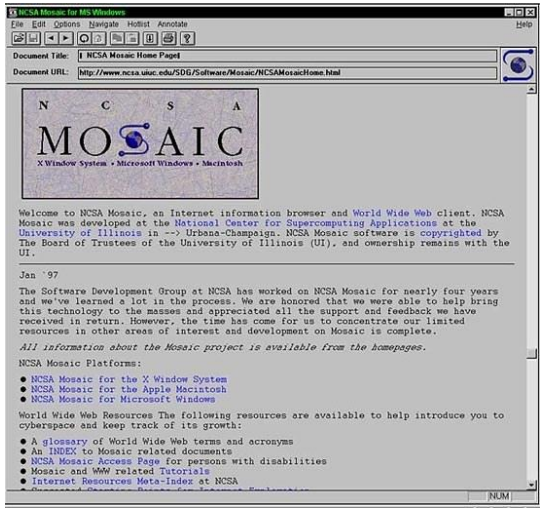
The World Wide Web

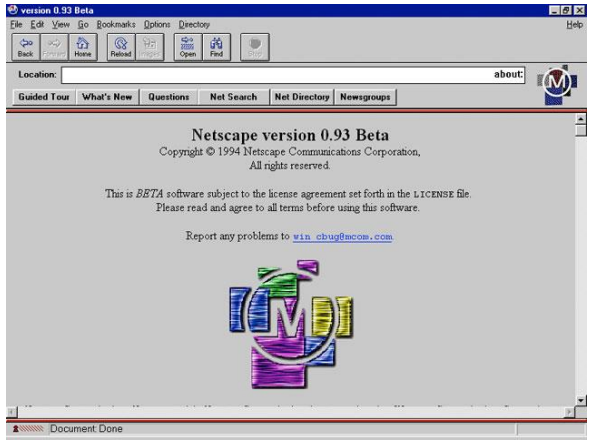
In 1989, Tim Berners-Lee invented a new language, HTML, and a new notation, URL, that would revolutionize how people communicated over the Internet. He wanted people to be able to share information more easily with each other.

This led to the beginning of websites as we know them. For this contribution, Tim Berners-Lee is known as the father of the World-Wide Web.

1990s: Web Browsers and Search Engines

With HTML came web browsers that could parse the HTML into structured text, to make it easier to read. Mosaic and Netscape Navigator were two of the first browsers.





Search engines also started popping up in the 1990s. Google wasn't founded until 1998, and Wikipedia wasn't created until 2001!

2000s: Social Media and Cloud Computing

As more and more people got on the internet, social media networks started to pop up. Some started in the late 90s; of the current big networks, LinkedIn started in 2003, Facebook in 2004, and Twitter in 2006.

Cloud Computing also started in the 2000s. Amazon's Elastic Compute Cloud started in 2006; Microsoft Azure started in 2008.

2010s: Smartphones and Tablets

The growth of the internet and the desire to remain connected led to portable computing devices.

Smartphones first appeared in 2007 with the release of the iPhone and gained widespread popularity in the 2010s. Tablets also became popular in this timeframe.

Who knows what revelations the next decade will bring?

Main Takeaways

- The internet started out as a government project (ARPANET); when the government left it, it became more public
- The invention of TCP/IP and HTML made the internet more widely accessible.
- The internet as we know it has changed drastically over the past few decades and will probably continue to change!

References

- Tom Cortina's class on Computing History: cs.cmu.edu/~15292/index.html

- And many other helpful webpages:
- explainthatstuff.com/historyofcomputers.html
- worldsciencefestival.com/infographics/a_history_of_computer_science/
- cs.uwaterloo.ca/~shallit/Courses/134/history.html
- wikipedia.org/wiki/History_of_computer_science