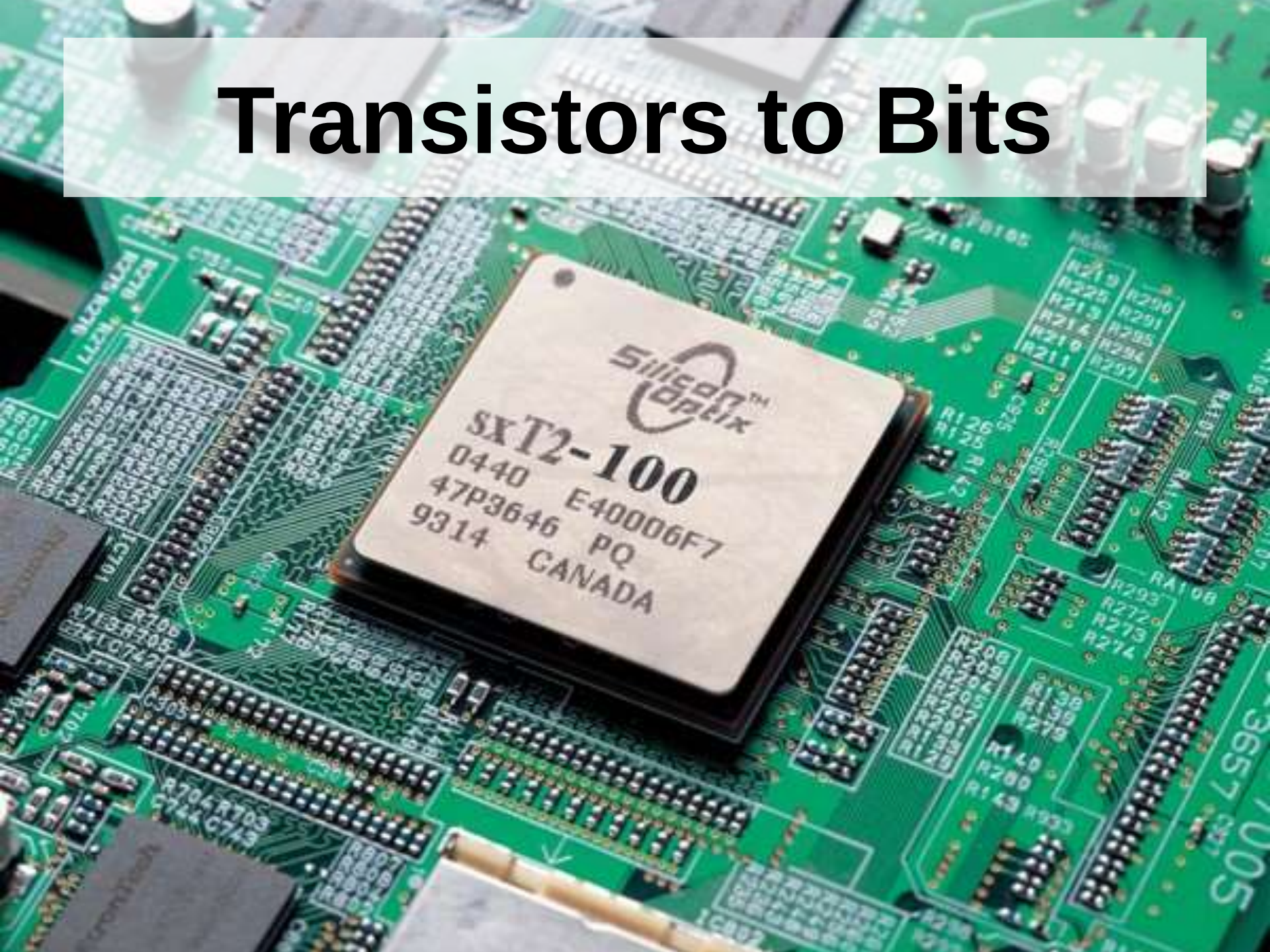
A Brief History of Programming

Jean Yang
Women's Coding Collective
December 3, 2014


iPad

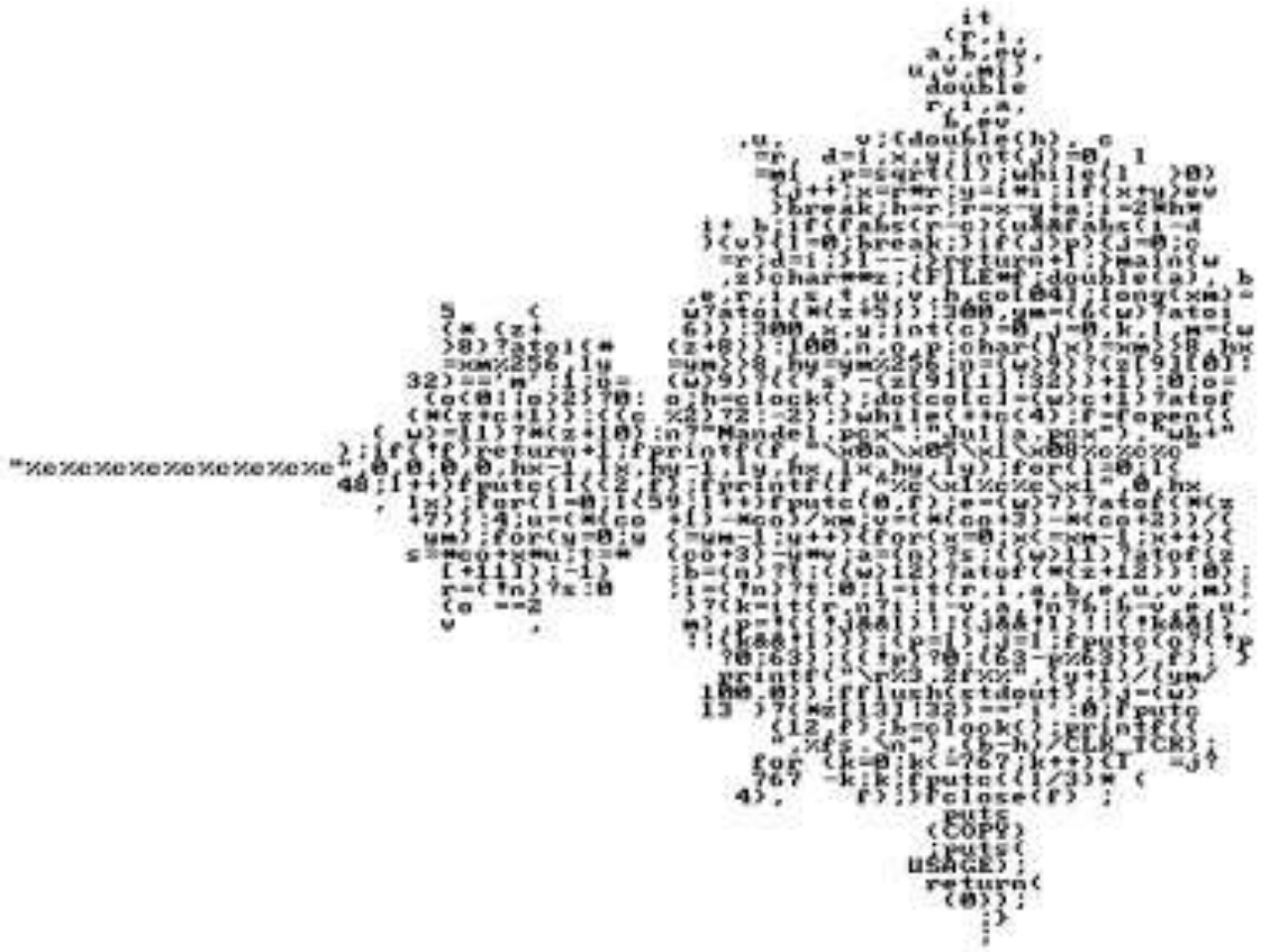
Transistors to Bits



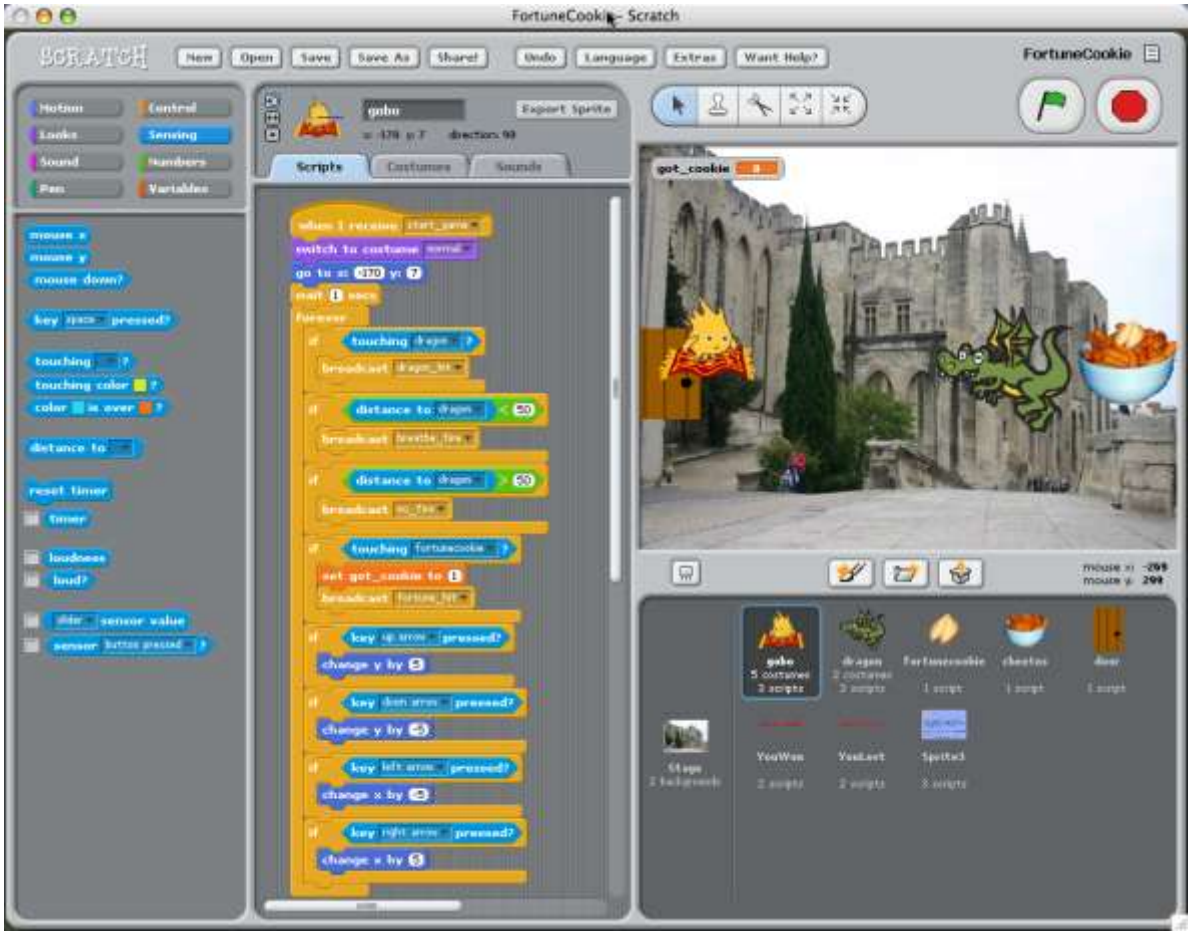
Binary Bits to Assembly

```
00000000                push    ebp
00000001                mov     ebp, esp
00000003                movzx   ecx, [ebp+arg_0]
00000007                pop     ebp
00000008                movzx   dx, cl
0000000C                lea     eax, [edx+edx]
0000000F                add     eax, edx
00000011                shl     eax, 2
00000014                add     eax, edx
00000016                shr     eax, 8
00000019                sub     cl, al
0000001B                shr     cl, 1
0000001D                add     al, cl
0000001F                shr     al, 5
00000022                movzx   eax, al
00000025                retn
```

Assembly to Languages



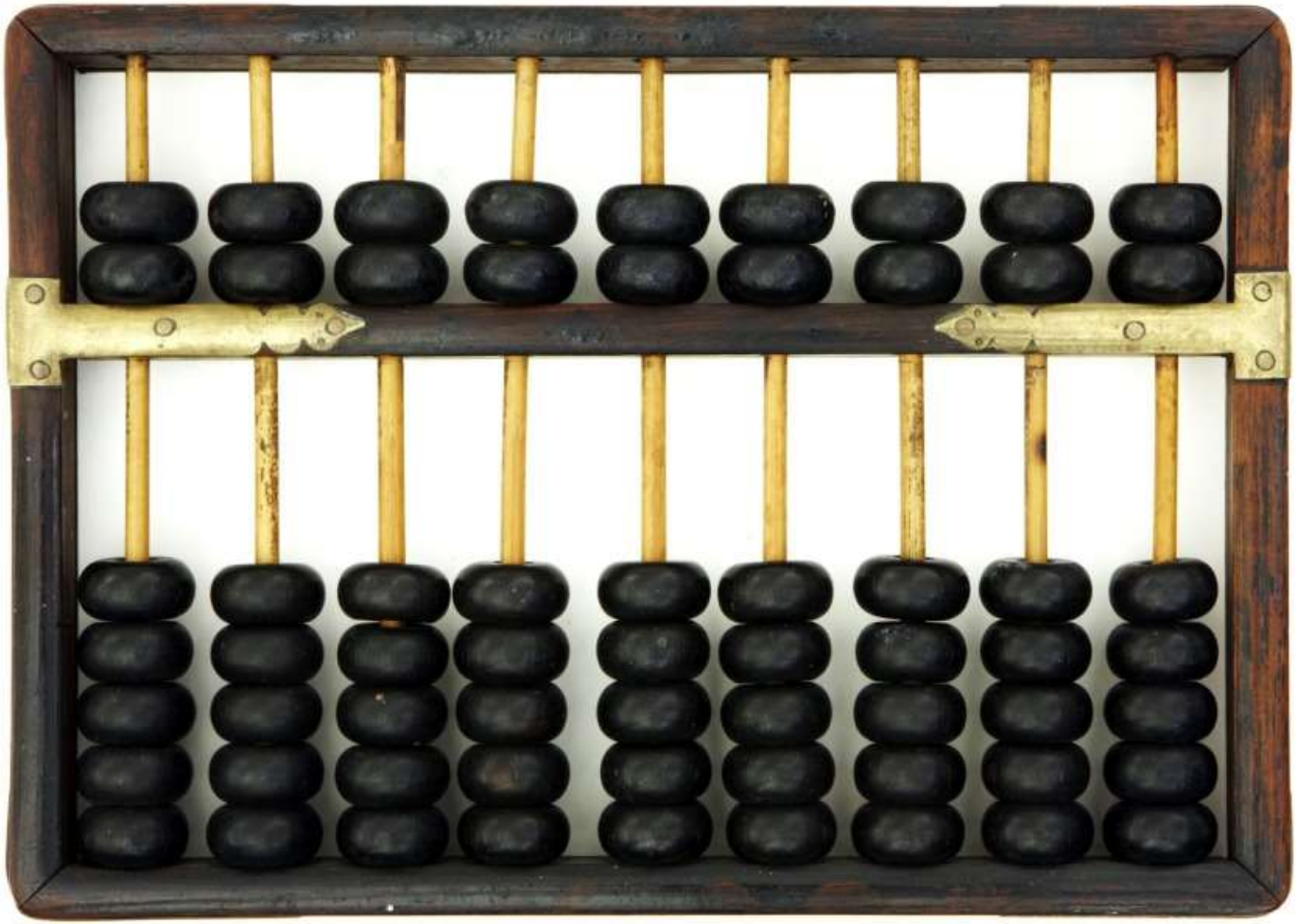
Languages to More Languages



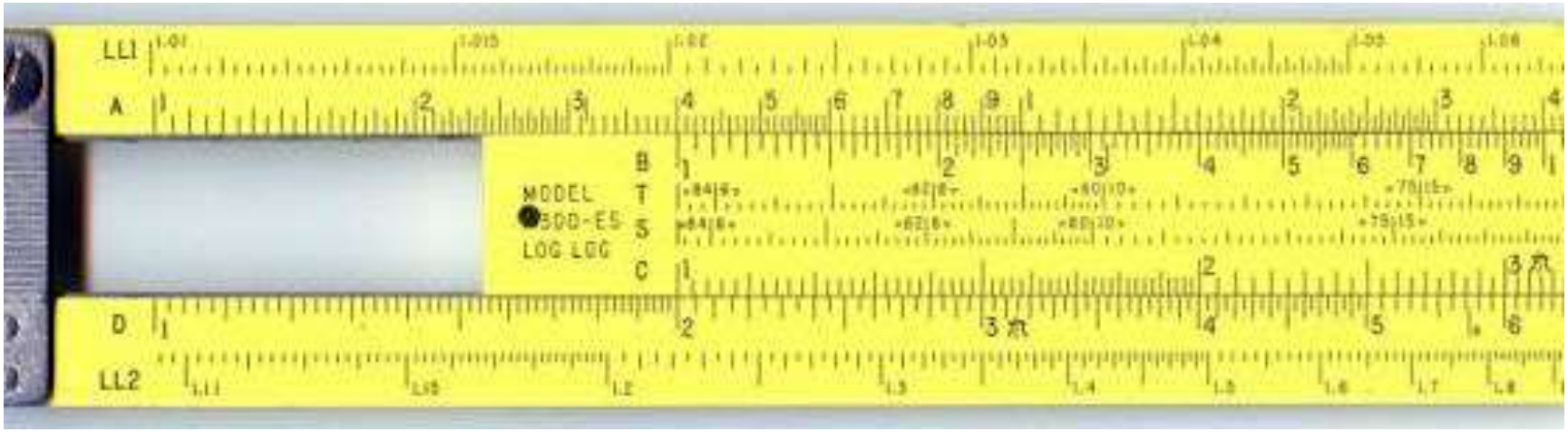

/www.etsy.com/shop/sharpwriter
NEUSER

Machines that Count

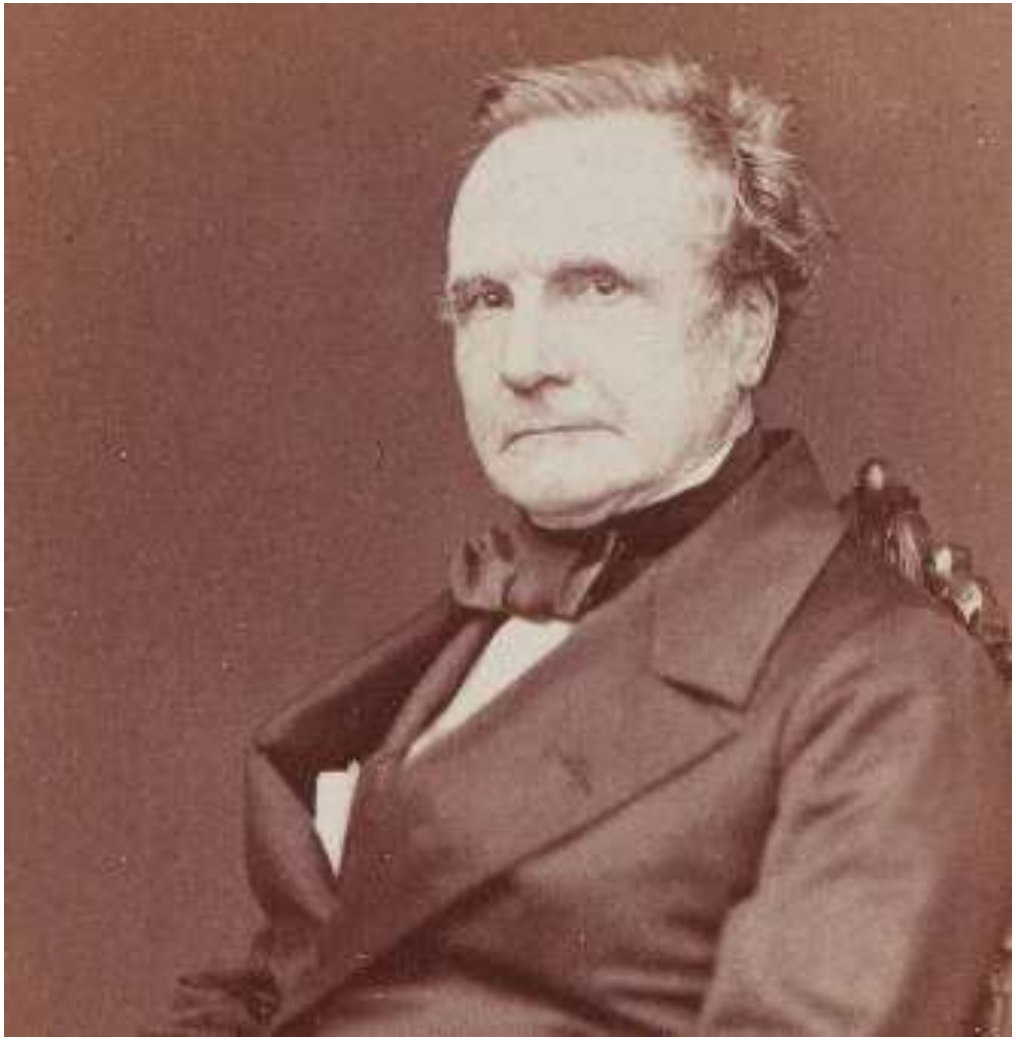
Abacus (2700-2300 BC)



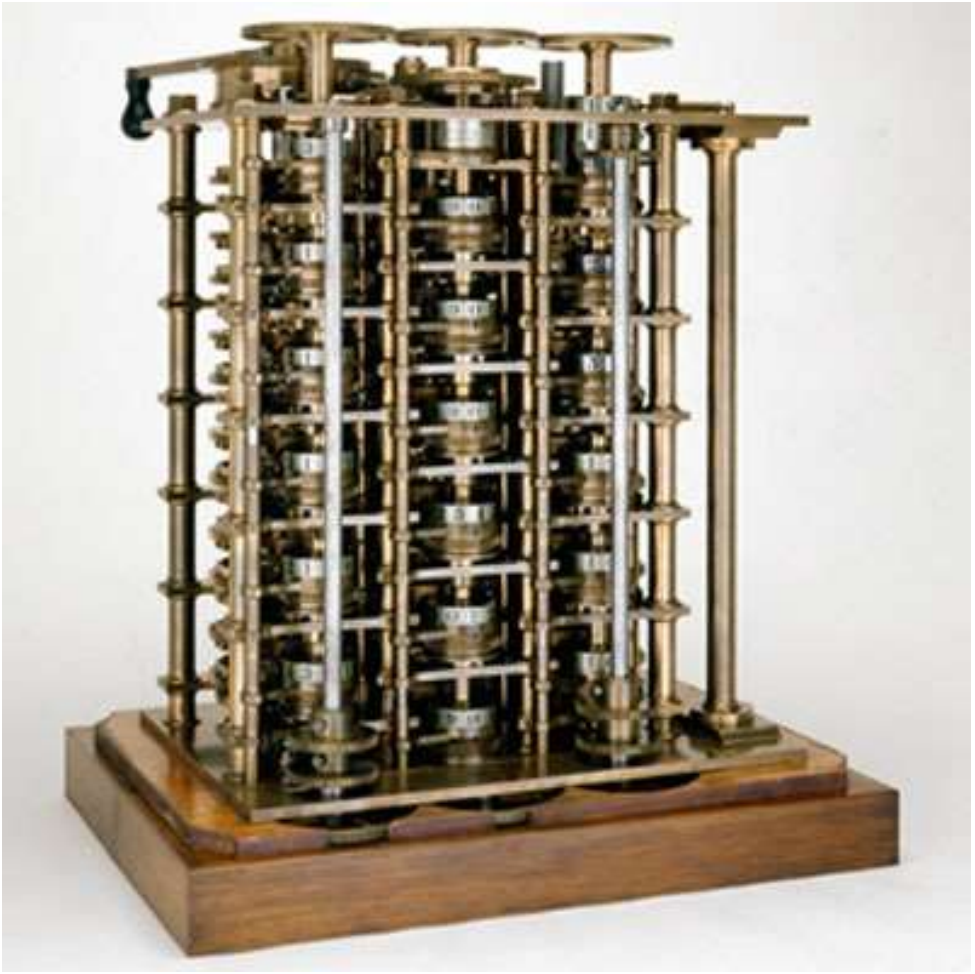
Slide Rule (17th century)



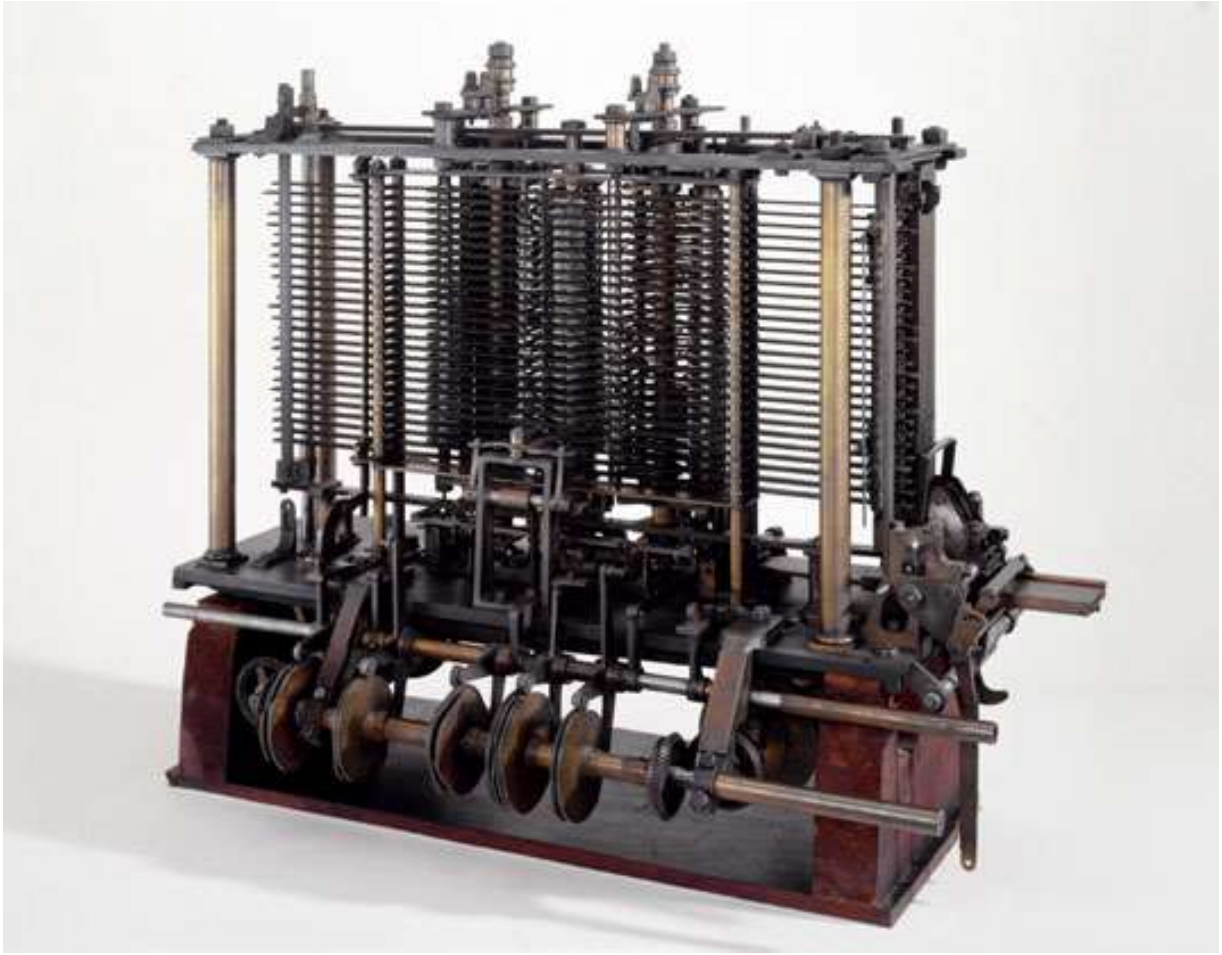
Charles Babbage



Difference Engine (1832)



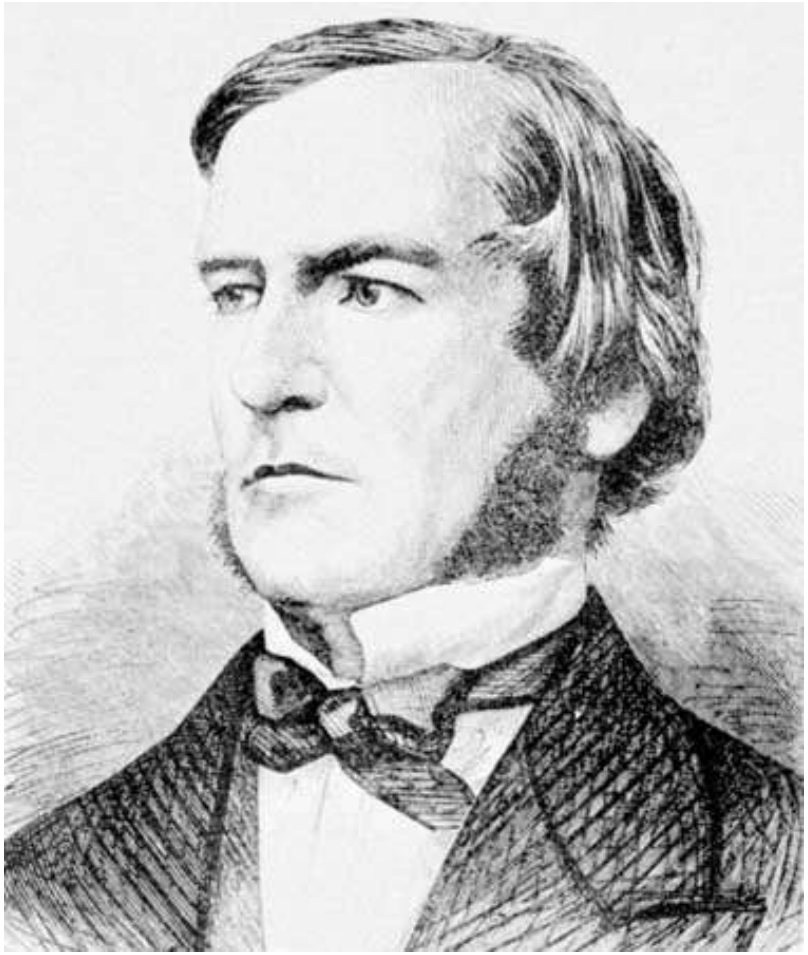
Analytical Engine (1834)



Ada Lovelace

Theory of Computers and Computation

George Boole



Boolean Algebra (1847)

TABLE 1-1 Basic Identities of Boolean Algebra

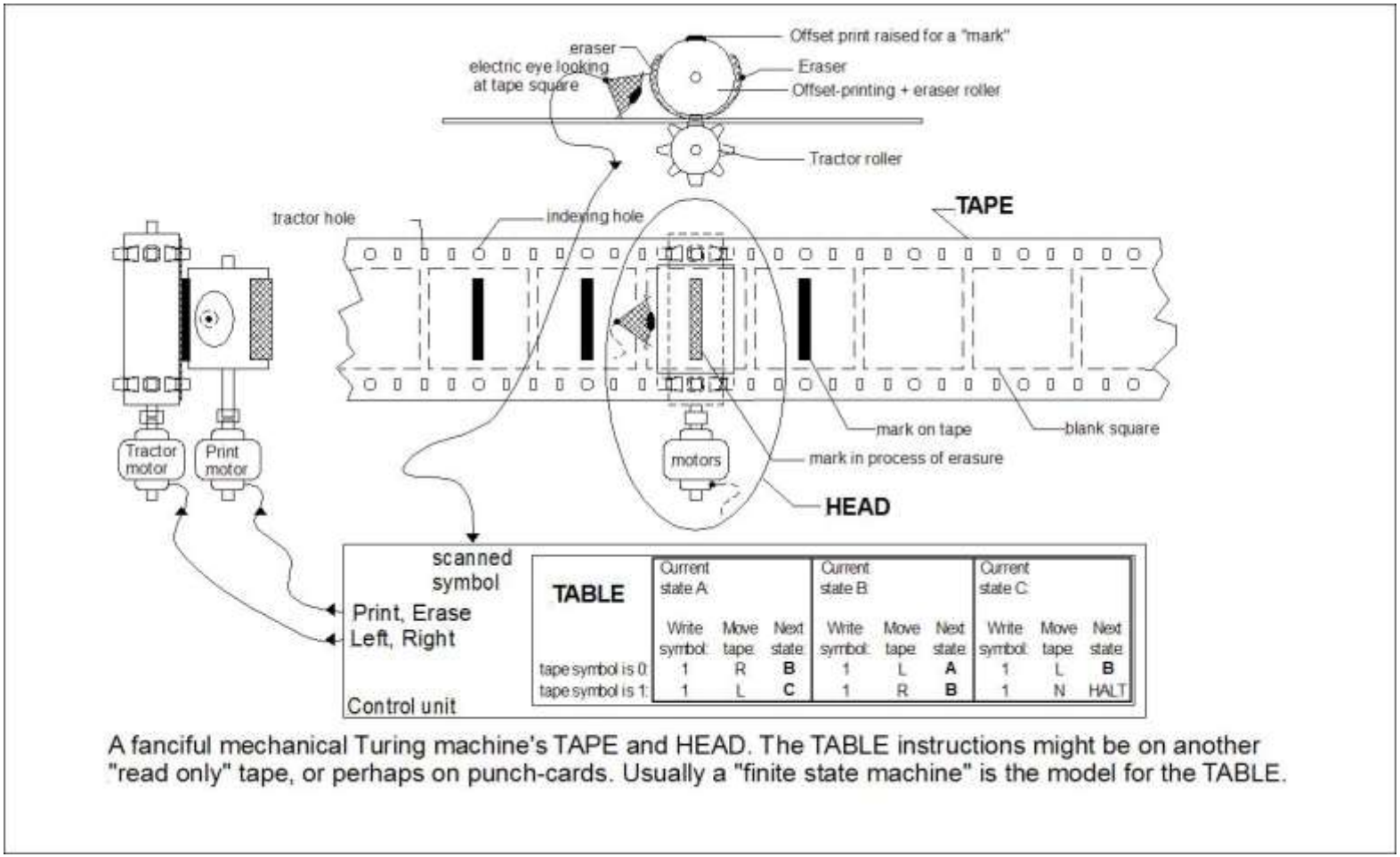
(1) $x + 0 = x$	(2) $x \cdot 0 = 0$
(3) $x + 1 = 1$	(4) $x \cdot 1 = x$
(5) $x + x = x$	(6) $x \cdot x = x$
(7) $x + x' = 1$	(8) $x \cdot x' = 0$
(9) $x + y = y + x$	(10) $xy = yx$
(11) $x + (y + z) = (x + y) + z$	(12) $x(yz) = (xy)z$
(13) $x(y + z) = xy + xz$	(14) $x + yx = (x + y)(x + z)$
(15) $(x + y)' = x'y'$	(16) $(xy)' = x' + y'$
(17) $(x')' = x$	

Lambda Calculus (1930s)

$$(\lambda y.\lambda x.y)\ z \quad \rightarrow \quad \lambda x.z$$

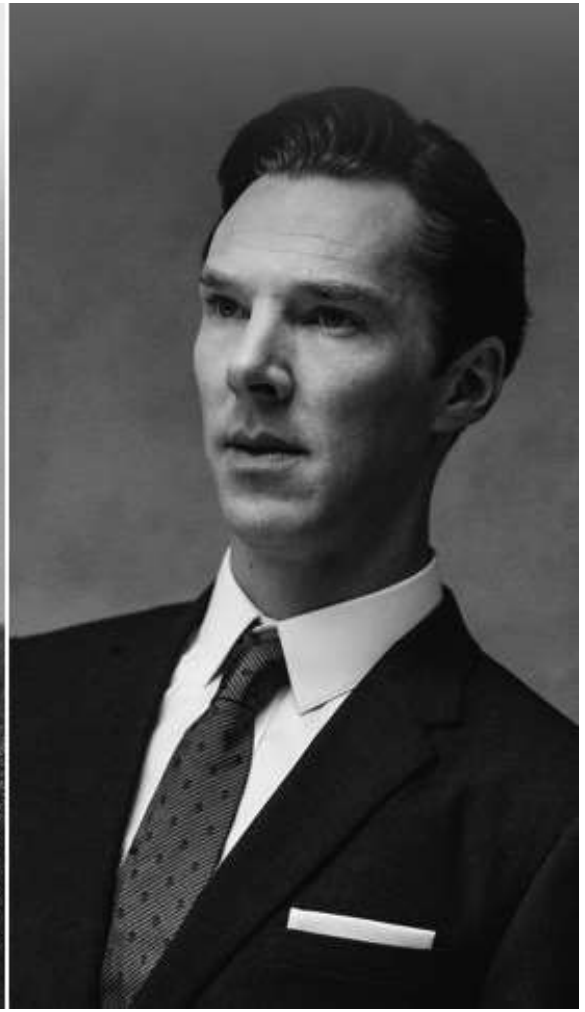
$$((y)\ \texttt{->}\ (x)\ \texttt{->}\ y)\ z \quad \rightarrow \quad (x)\ \texttt{->}\ z$$

Turing Machines (1936)



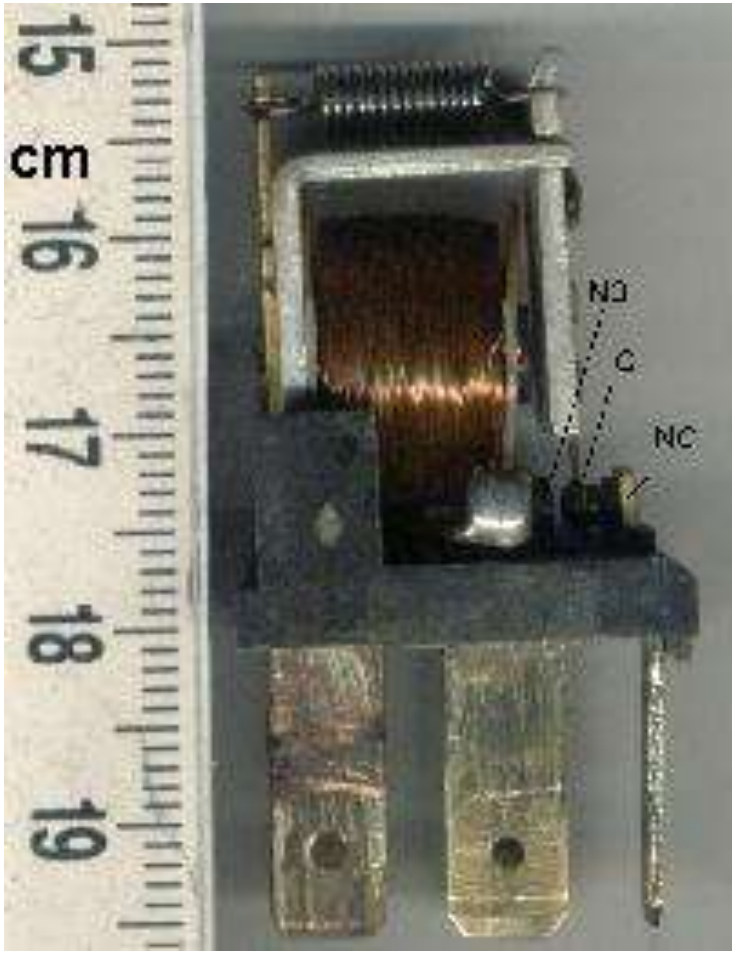
A fanciful mechanical Turing machine's TAPE and HEAD. The TABLE instructions might be on another "read only" tape, or perhaps on punch-cards. Usually a "finite state machine" is the model for the TABLE.

Alan Turing

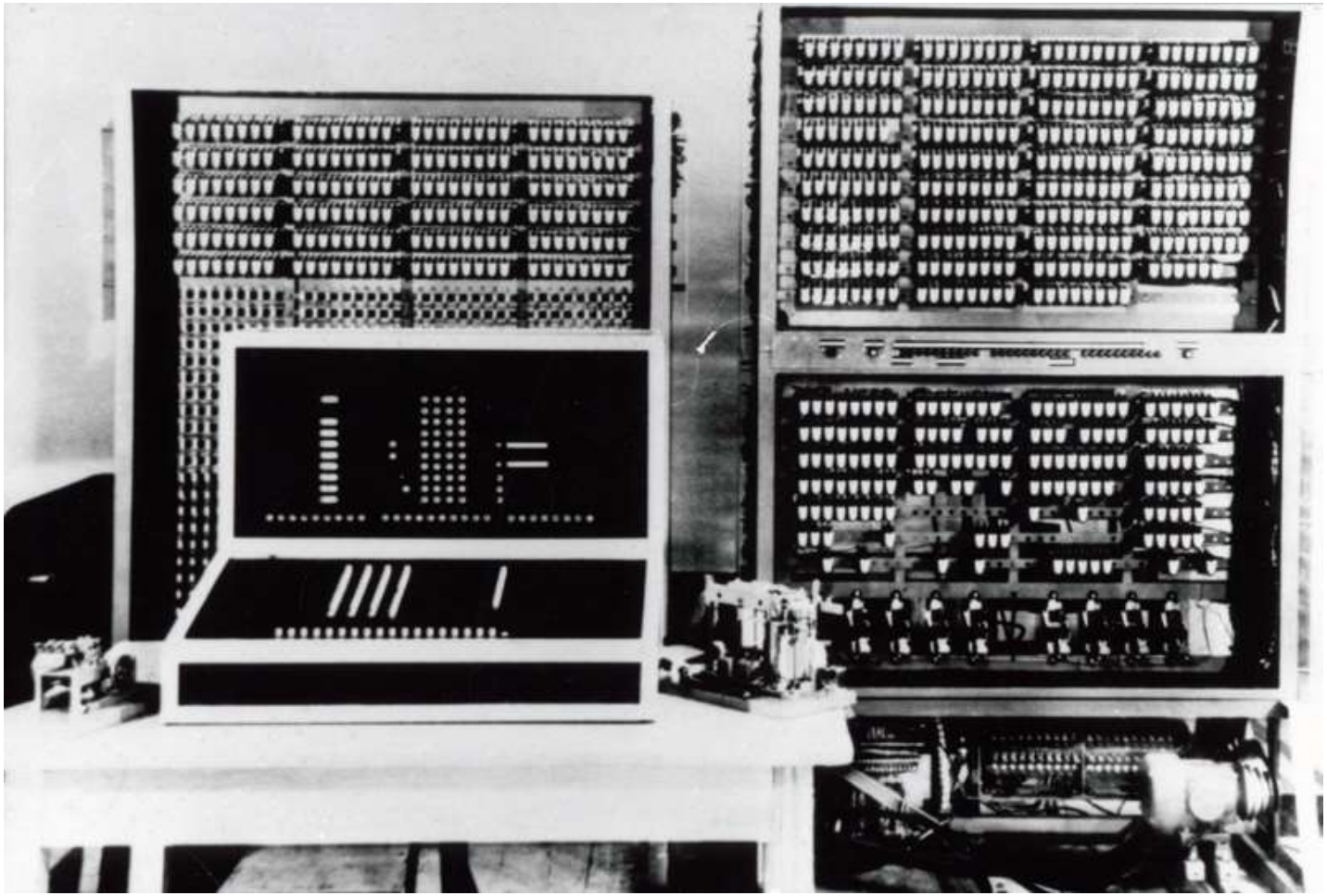


Towards Modern Computers

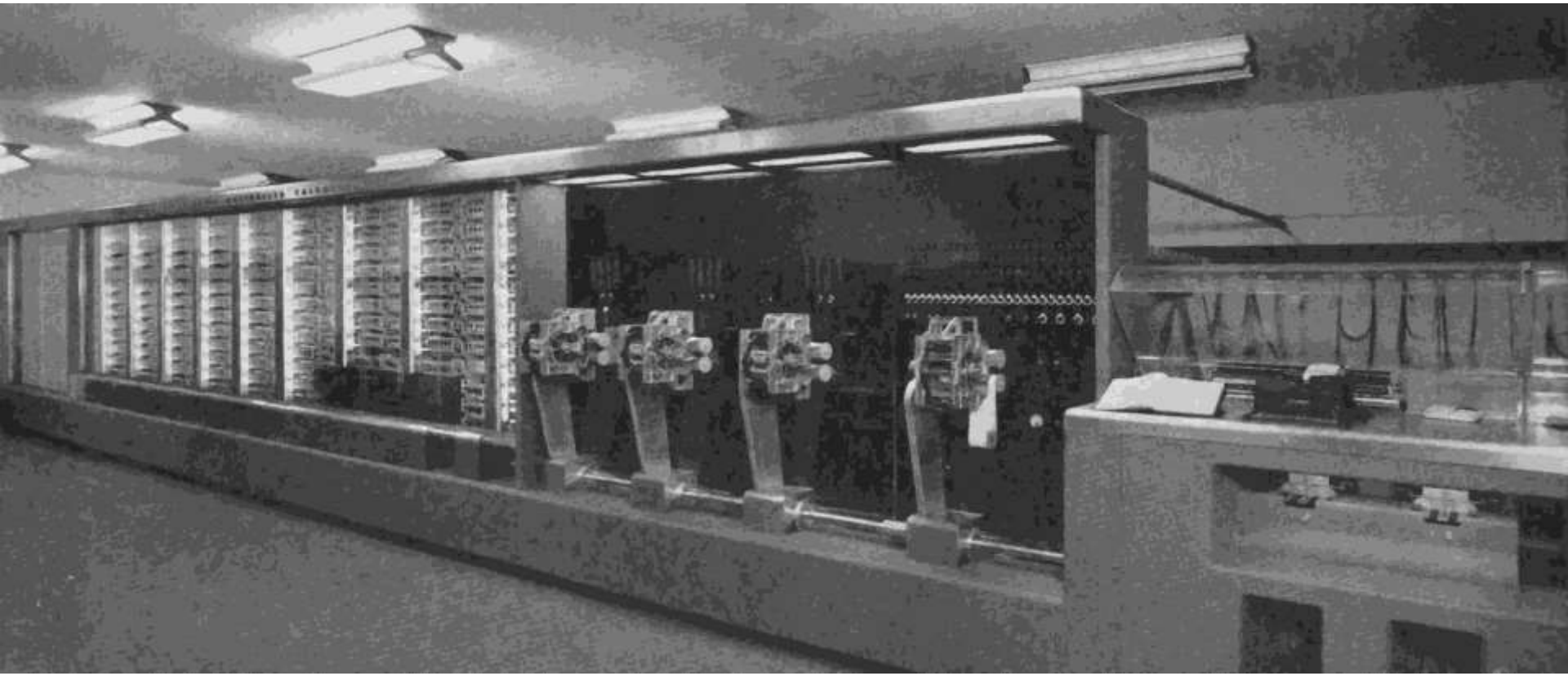
Relays



Z3 (1941)



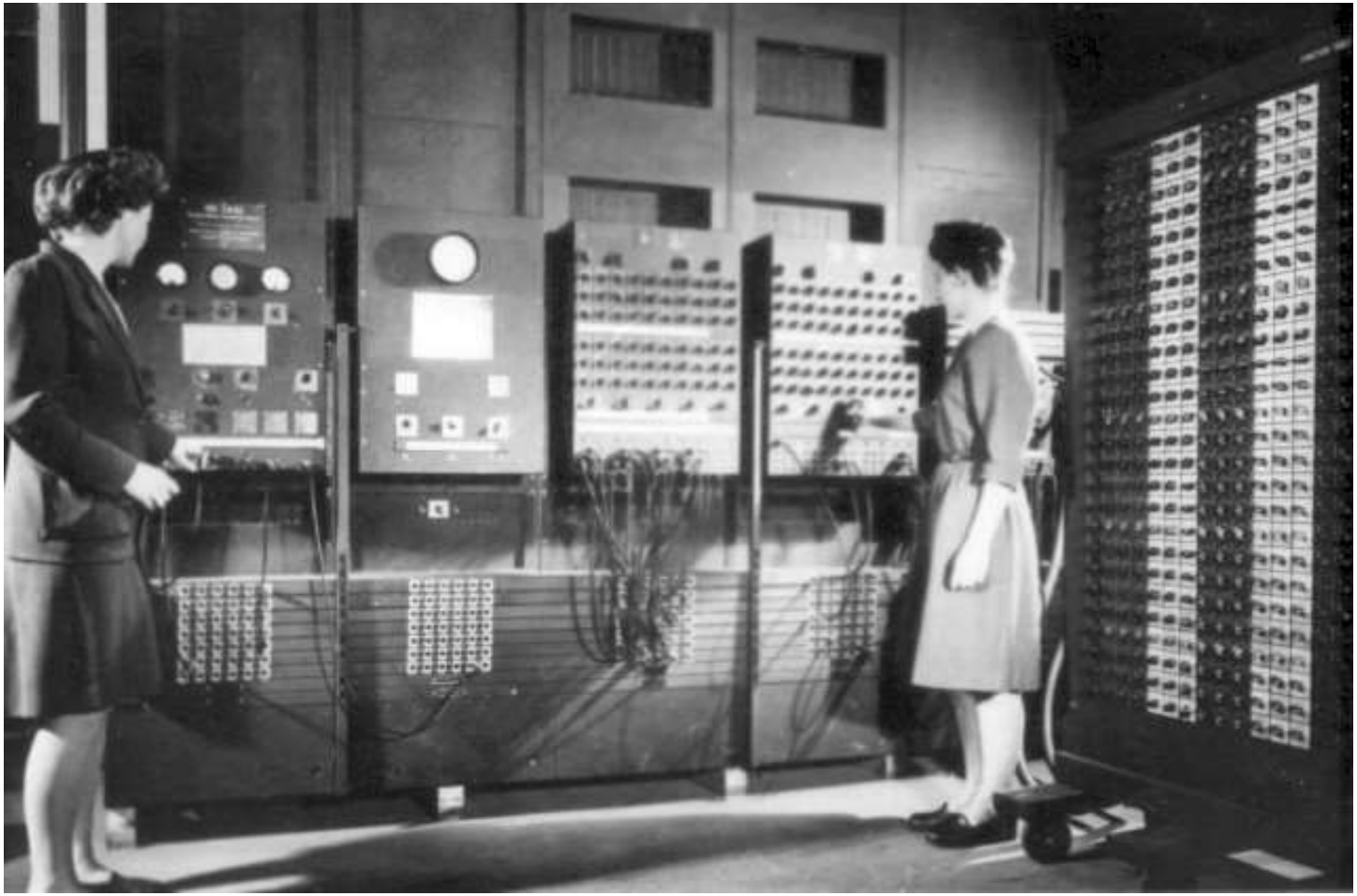
Mark 1 (1944)



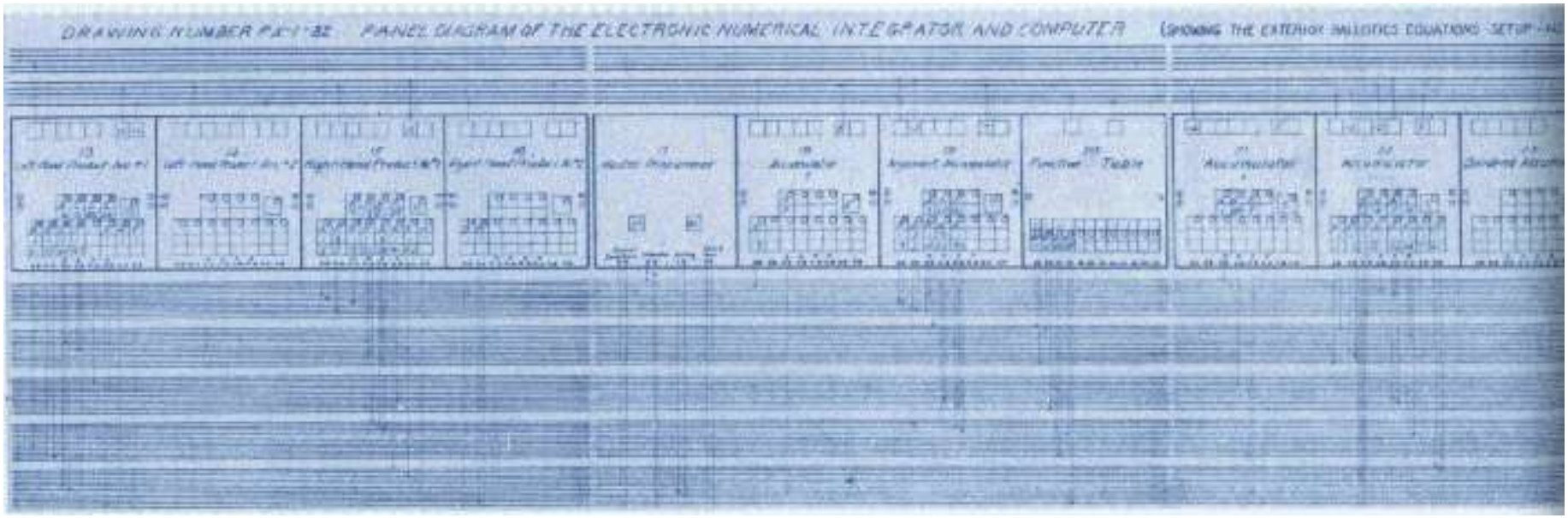
Grace Hopper

Vacuum Tube (1910)

ENIAC (1946)



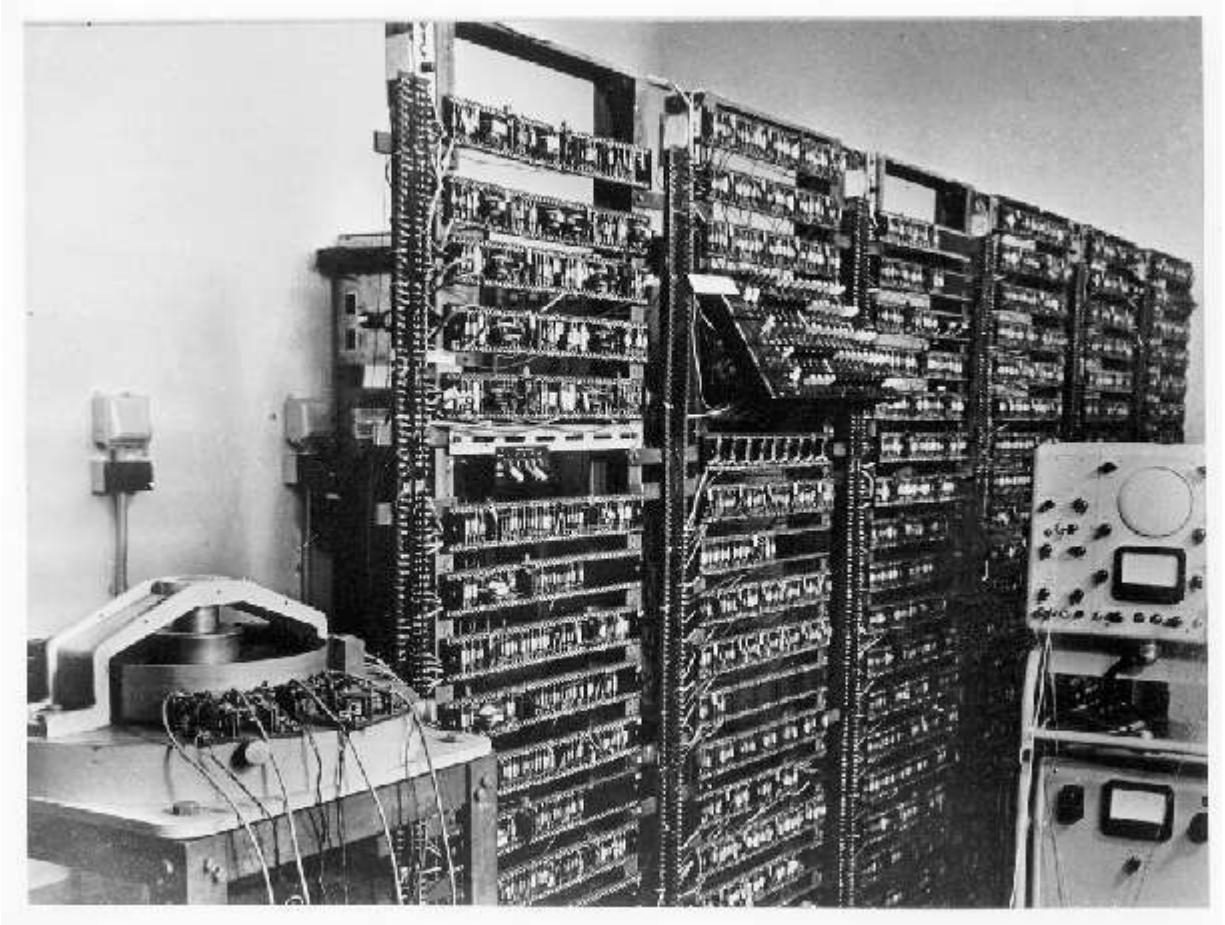
Ballistics Equations



Transistor (1947)

John Bardeen, William Shockley, and Walter Brittain

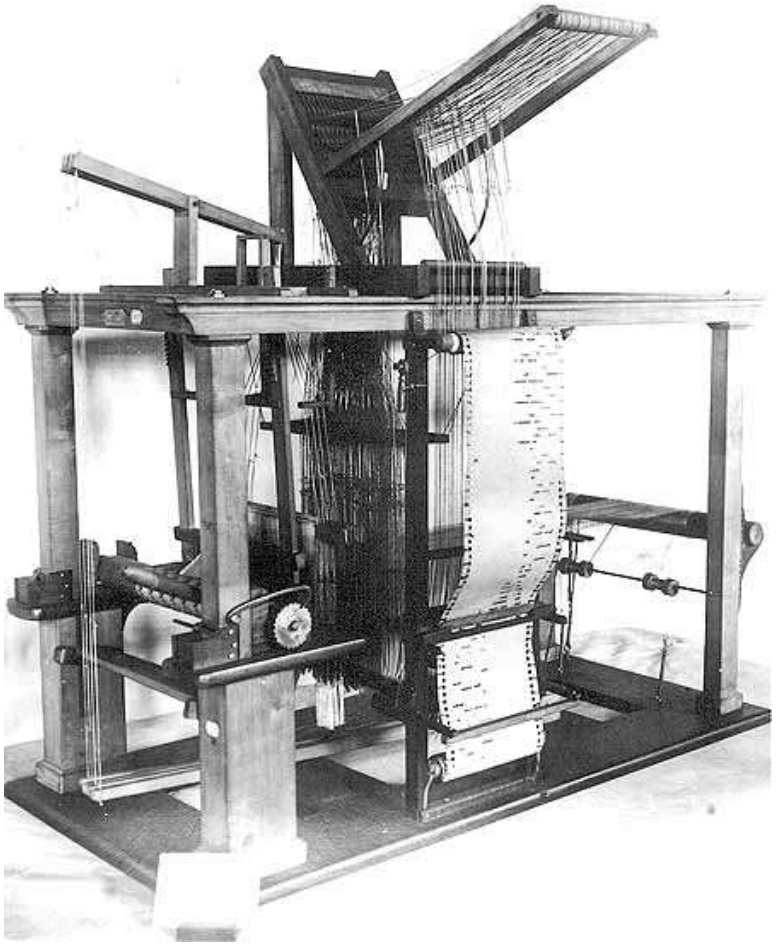
Transistor Computer (1953)



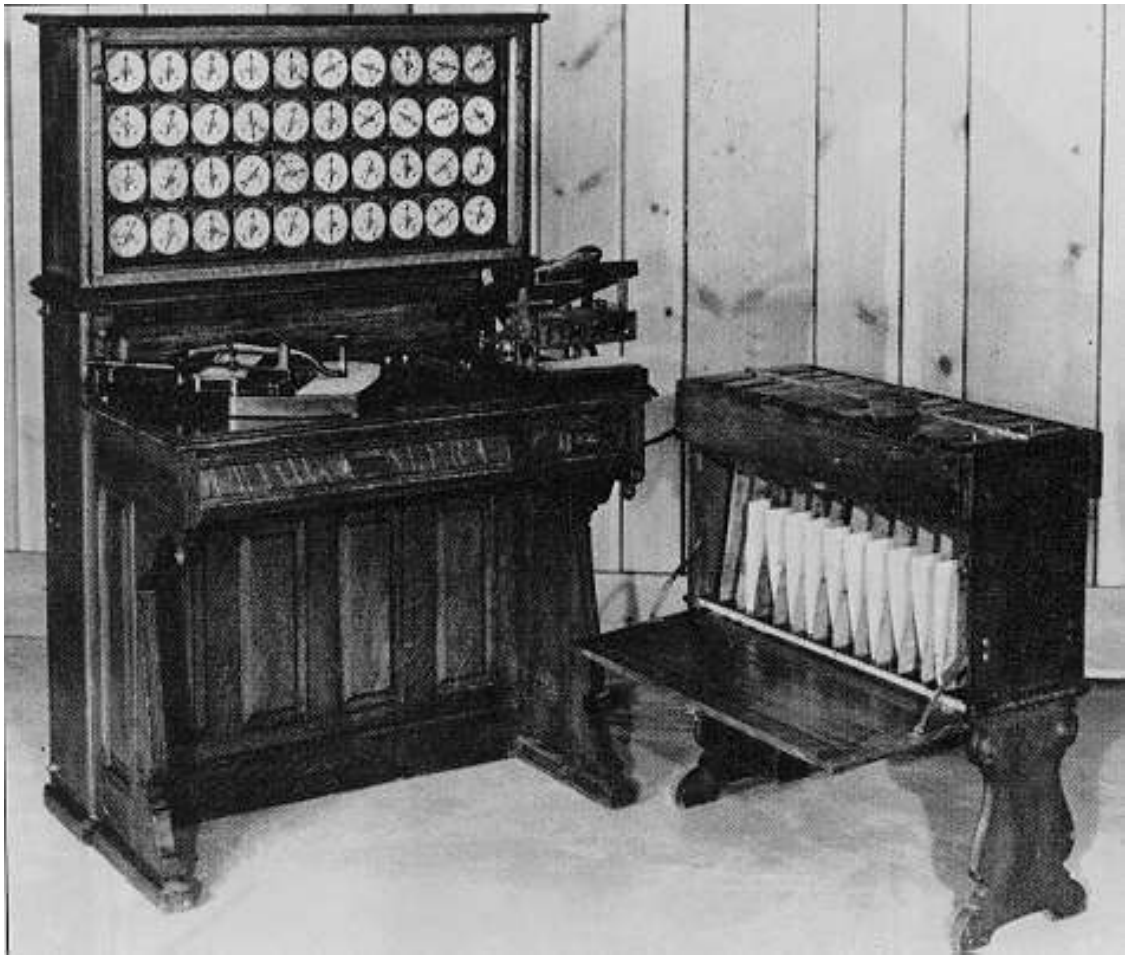
The Cards with the Holes

Joseph Marie Jacquard

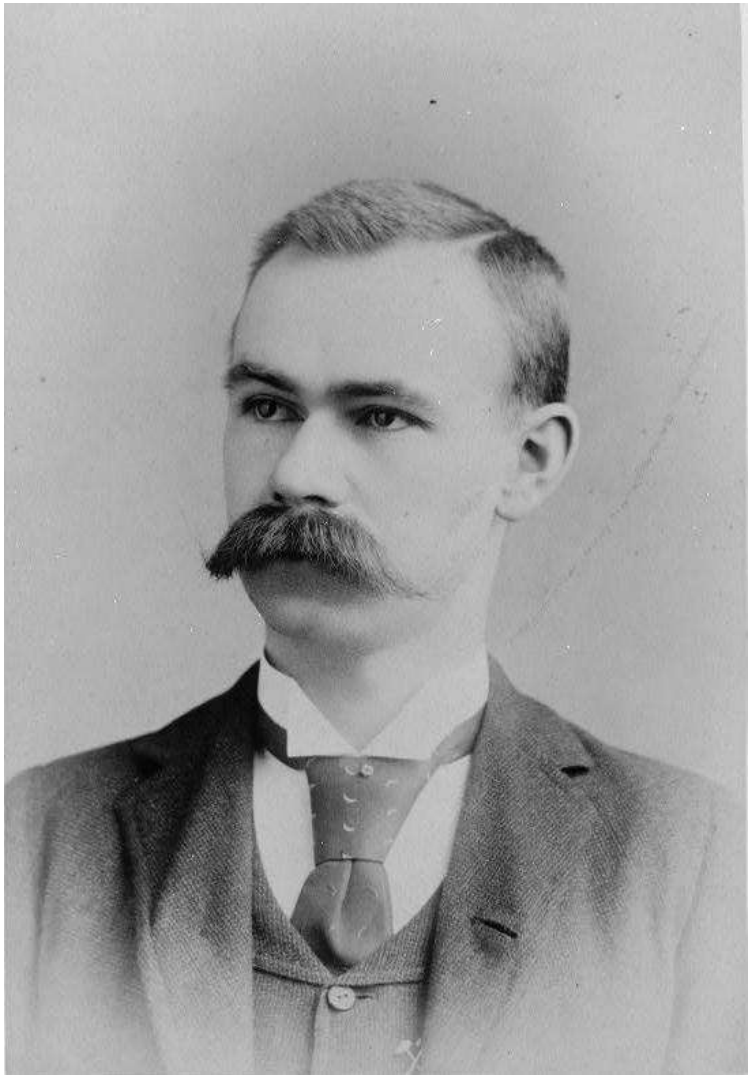
Jacquard Loom (1801)



Census Machine (1890)



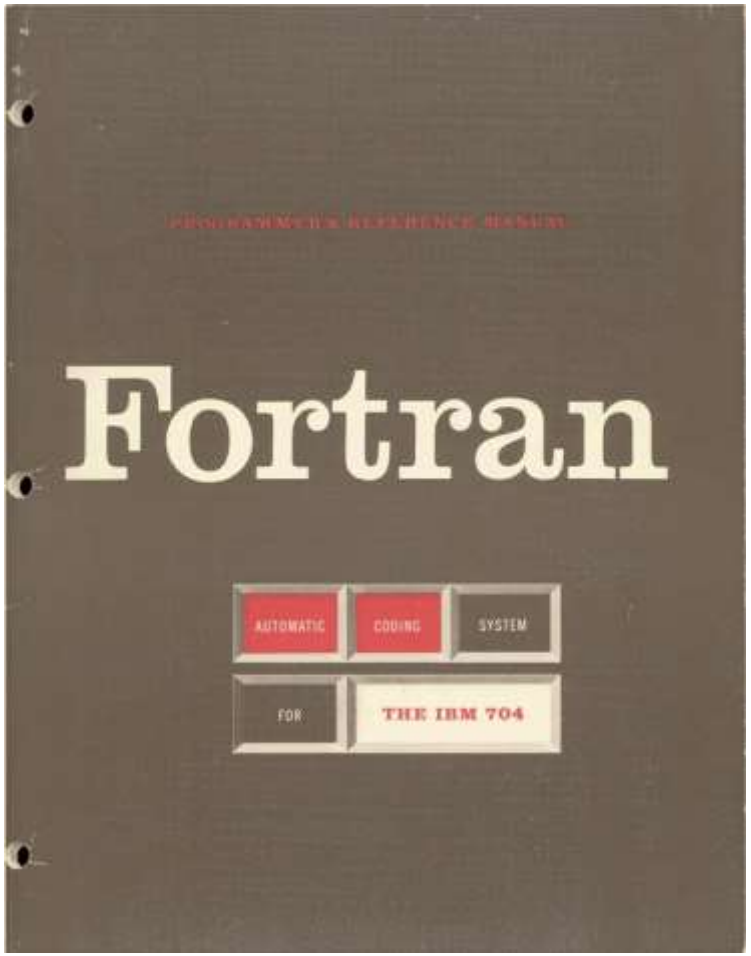
Herman Hollerith



IBM

Towards Modern Languages

FORTRAN (1950s)



BUSINESS INTERNATIONAL MACHINES

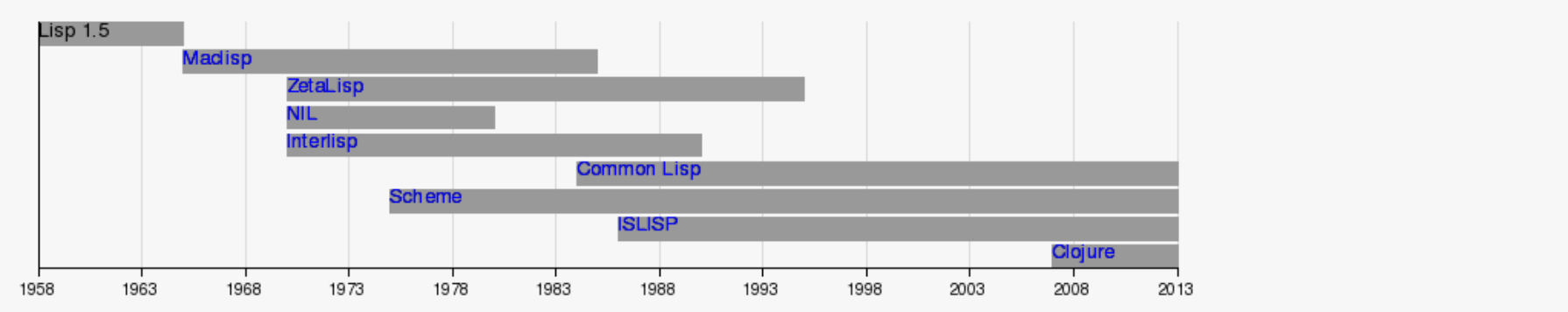
IBM

Fortran “Hello World”

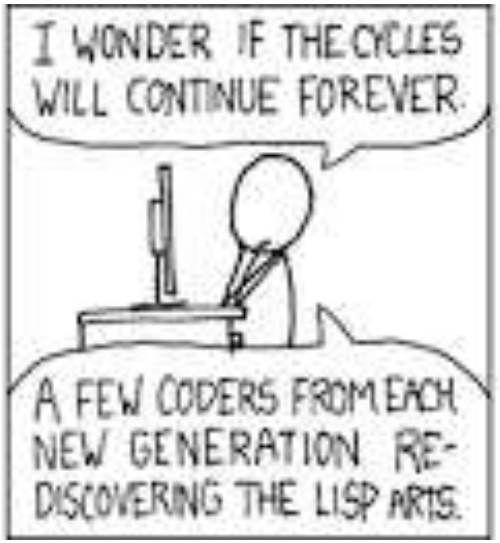
```
C      FORTRAN IV WAS ONE OF THE FIRST PROGRAMMING
C      LANGUAGES TO SUPPORT SOURCE COMMENTS
       WRITE (6,7)
          7  FORMAT(13H HELLO, WORLD)
       STOP
       END
```

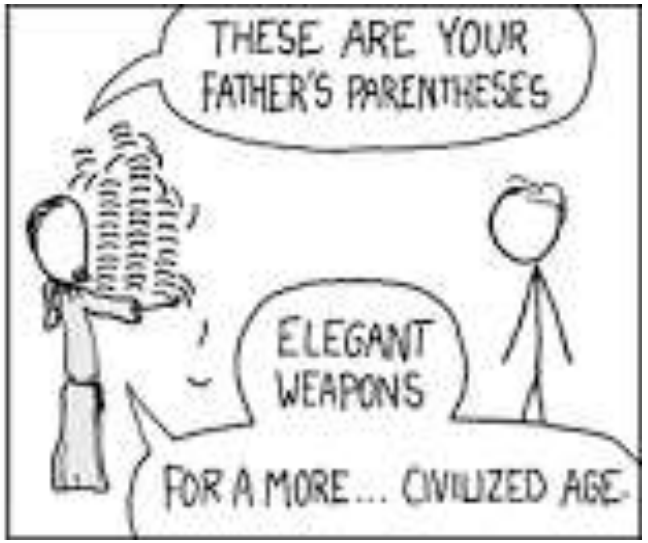
Fran Allen

Lisp (1958)


Lisp 1.5
Madlisp
ZetaLisp
NIL
Interlisp
Common Lisp
Scheme
ISLISP
Clojure
1958
1963
1968
1973
1978
1983
1988
1993
1998
2003
2008
2013


LISP IS OVER HALF A CENTURY OLD AND IT STILL HAS THIS PERFECT, TIMELESS AIR ABOUT IT.

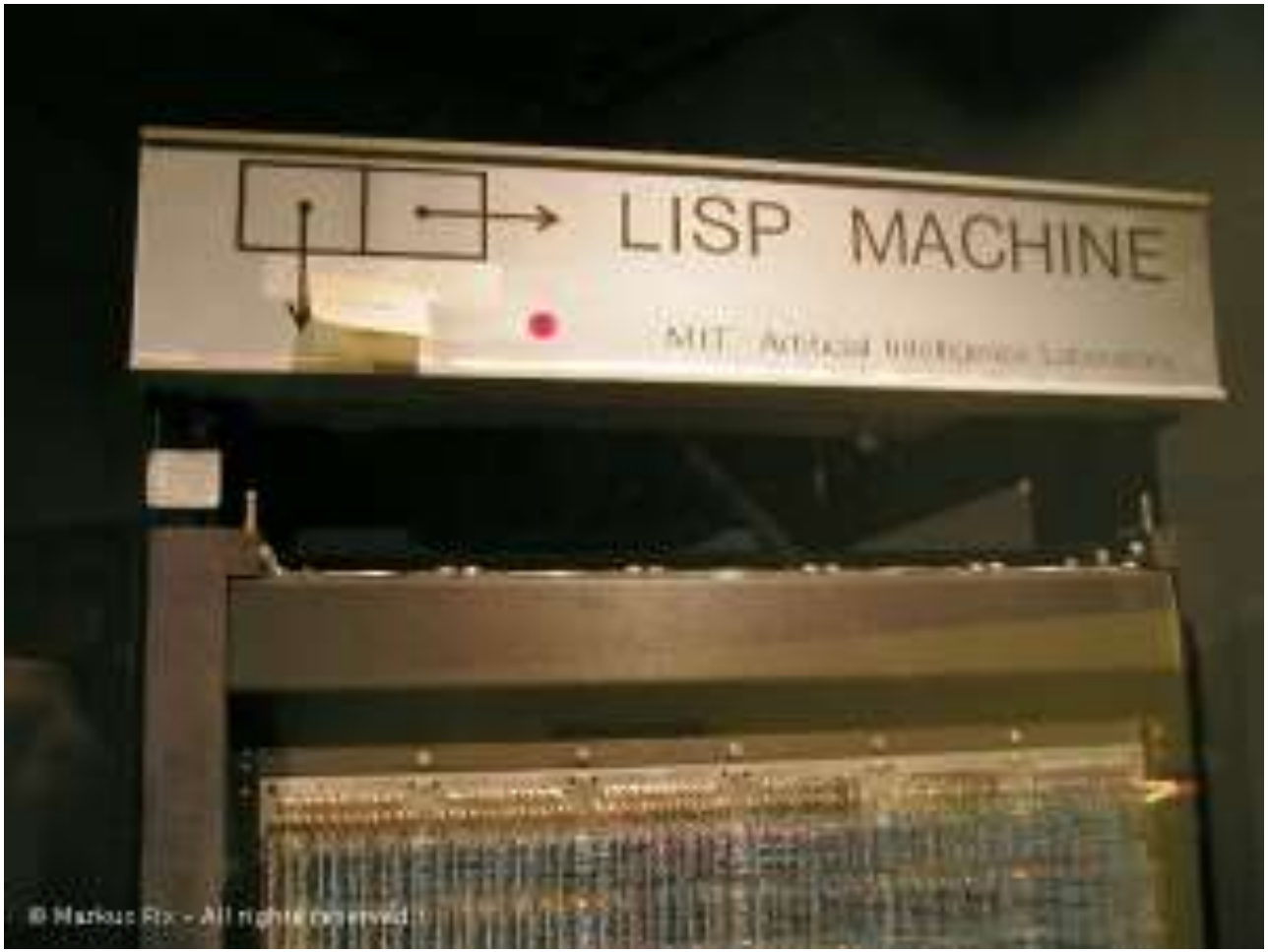

I WONDER IF THE CYCLES WILL CONTINUE FOREVER.
A FEW CODERS FROM EACH NEW GENERATION RE-DISCOVERING THE LISP ARTS.


THESE ARE YOUR FATHER'S PARENTHESES
ELEGANT WEAPONS
FOR A MORE... CIVILIZED AGE.

Lisp Program

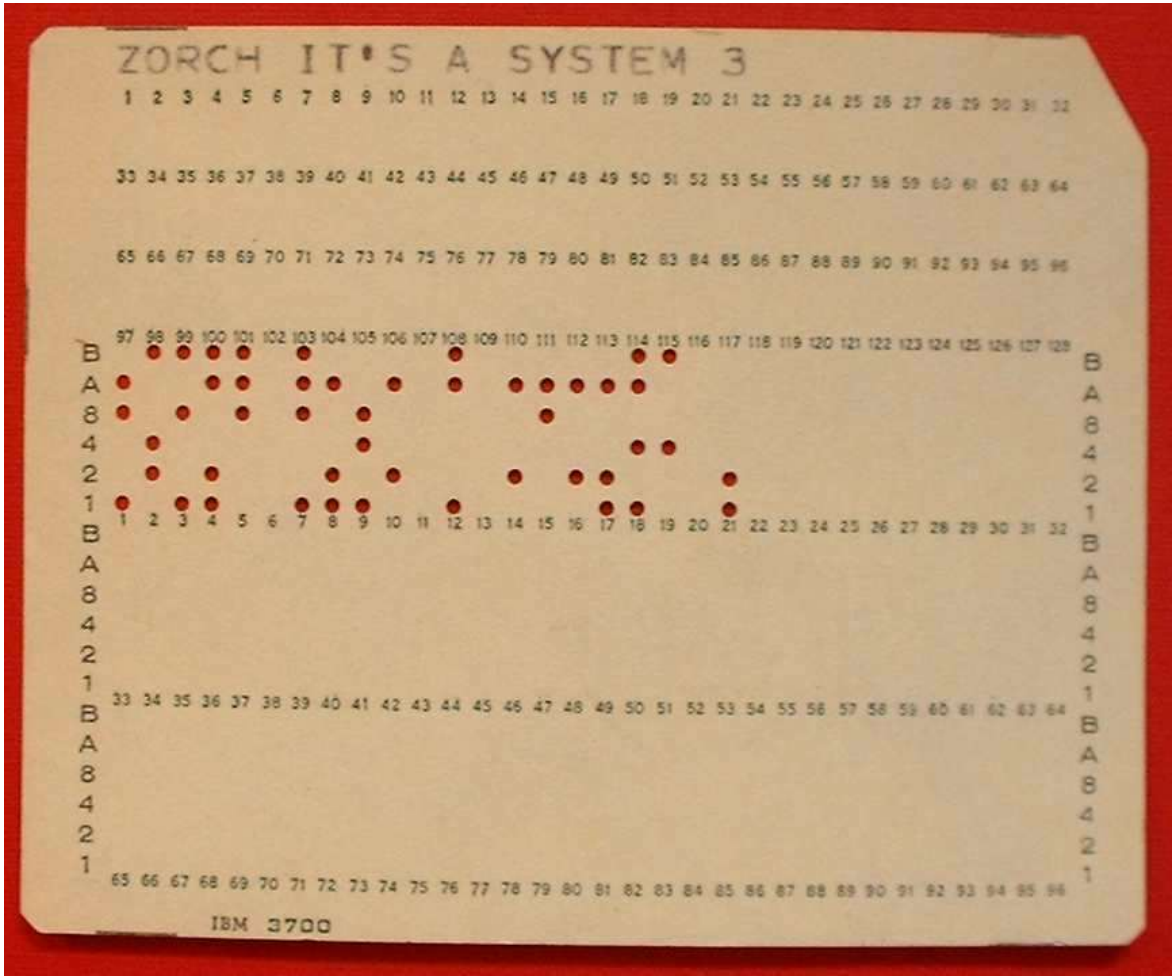
```
(defun factorial (N)
  "Compute the factorial of N."
  (if (= N 1)
    1
    (* N (factorial (- N 1)))))
```

Lisp Machines (1980s)

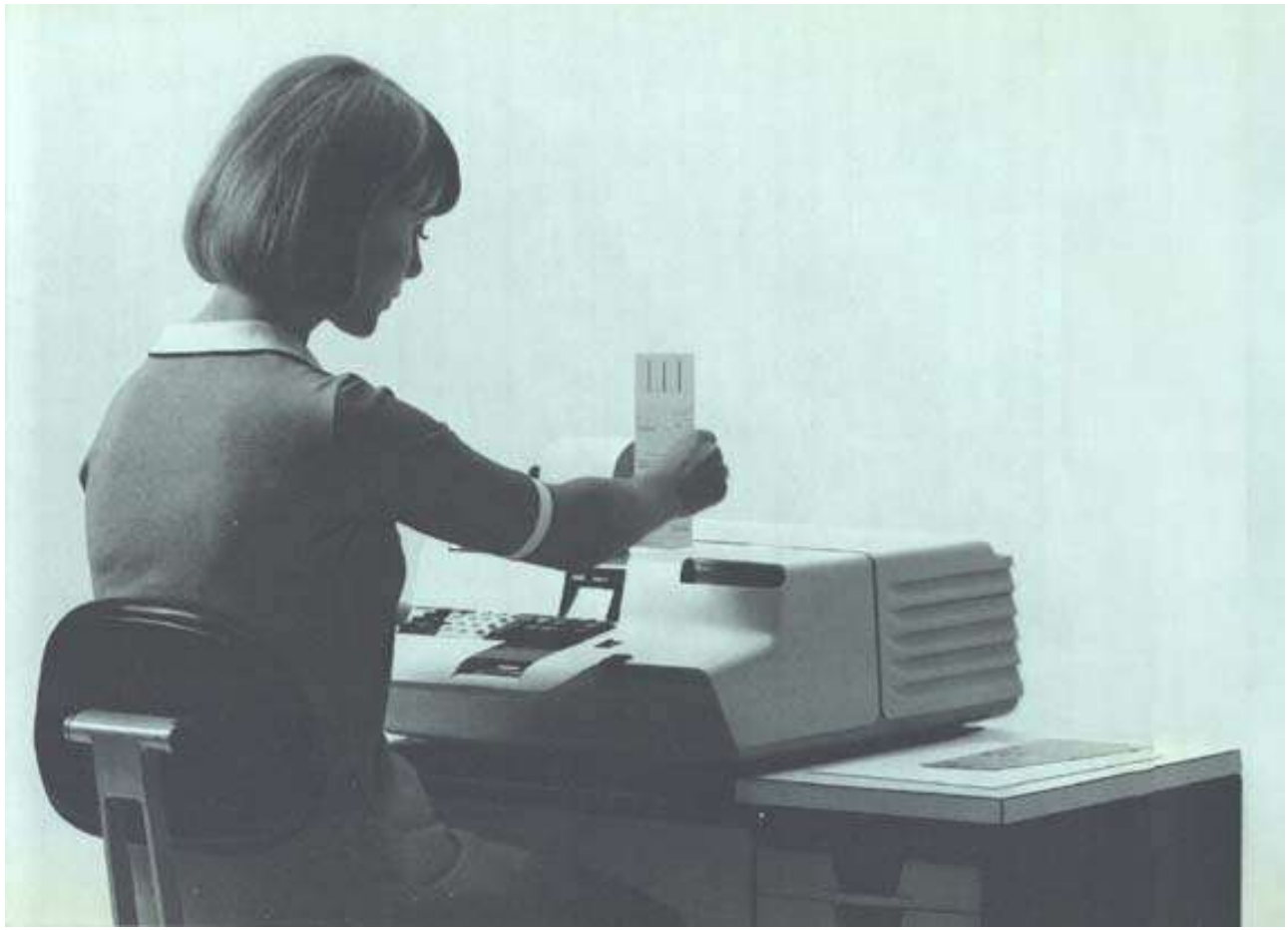


IBM 440 (1962)

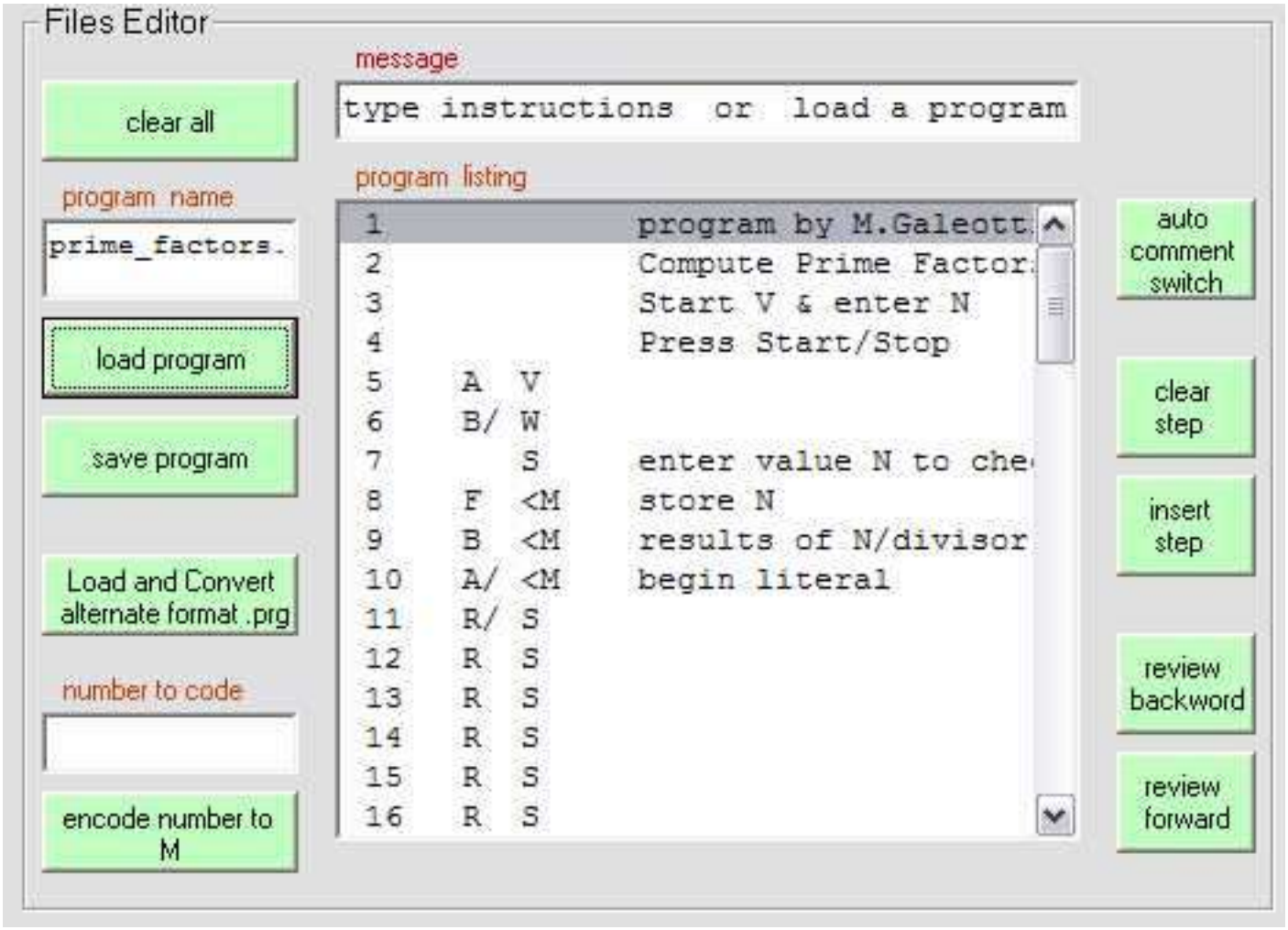
Punch Cards



Programma 101 (1965)



Programma Simulator



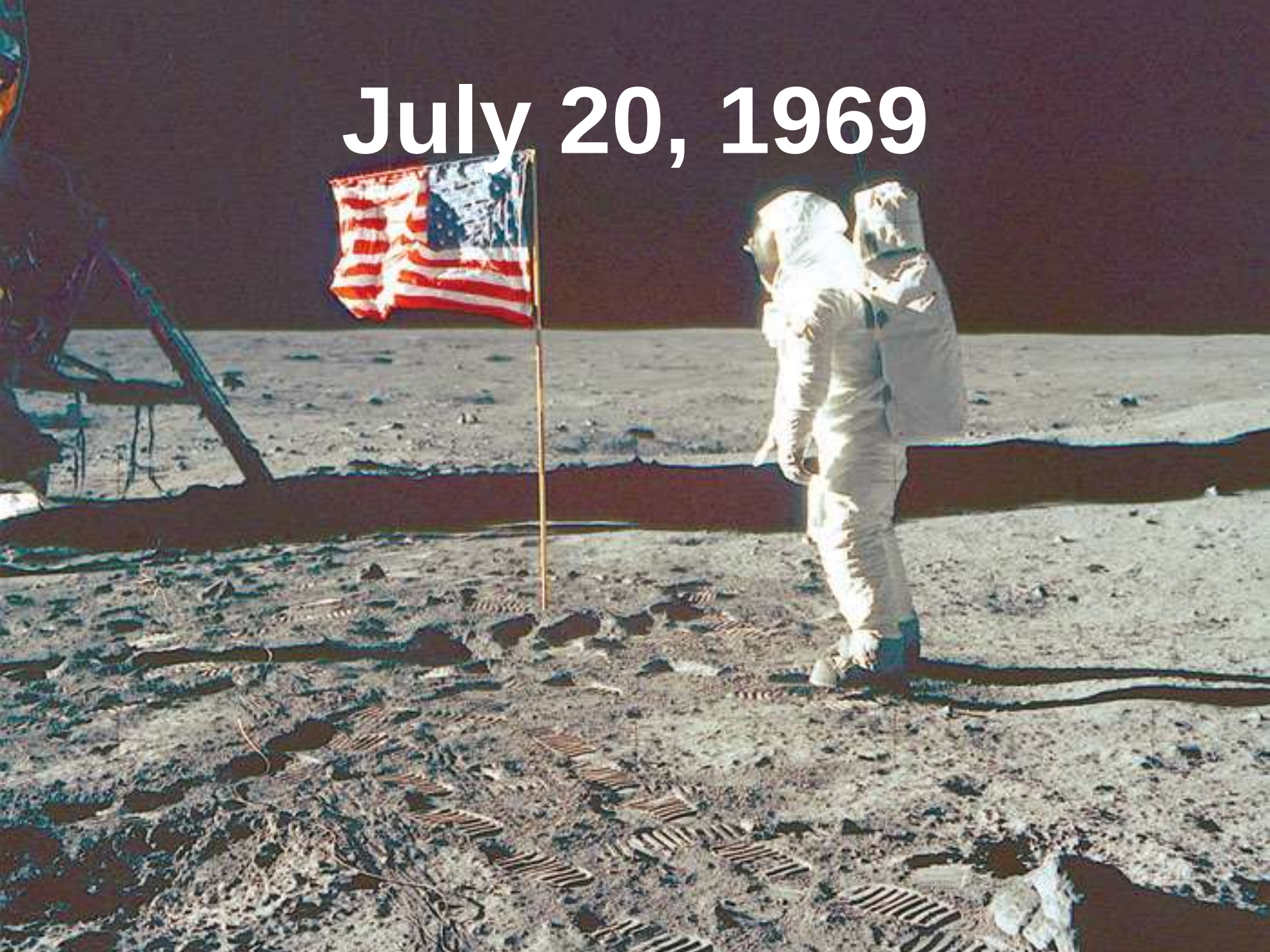
C (1969)

C “Hello World”

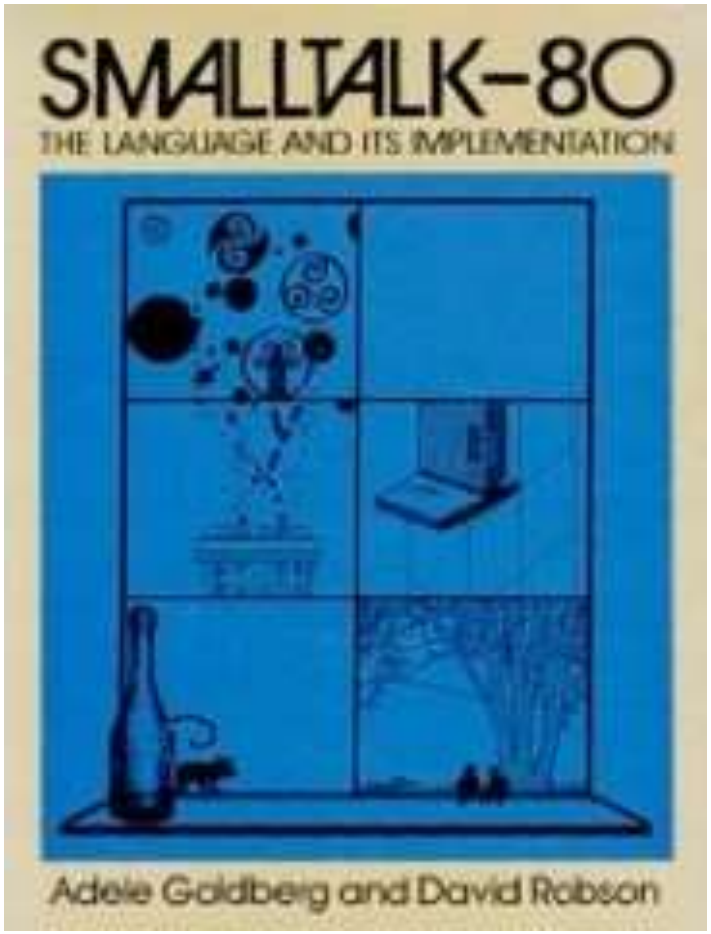
```
main() {
    printf("hello world");
}
```

July 3, 1969


INTERFACE
MESSAGE
PROCESSOR
Developed for
the Advanced Research Projects Agency
by Bolt Beranek and Newman Inc.
RESET
MASTER
CLEAR
START


July 20, 1969

Smalltalk (1970s)



Smalltalk Syntax

```
7     "a number"
$z    "a character"
'colourless ideas sleep furiously' "a string"
#(#tom #dick #harry)     "an array of 3 components"
#(# one 'should shower at least' 3 'times a week')
```

CLU (1970s)


Jordan Naoum (Ed.)
CLU (programming language)
Programming language, Massachusetts Institute of Technology, Barbara Jane Liskov
High Quality Content by WIKIPEDIA articles!
Duc

ML (1970s)

```
fun append (xs, ys) =
    if null xs
    then ys
    else (hd xs):: append (tl xs, ys)

fun map (f, xs) =
    case xs of
        [] => []
      | x :: xs' => (f x)::(map (f, xs'))

val a = map (increment, [4,8,12,16])
val b = map (hd, [[8,6],[7,5],[3,0,9]])
```

python (1989)

Java (1995)

Javascript (1995)

Swift (2014)

My Research: The Jeeves Language


```
findAllUsers(Harvard)
```

You have no friends in this location.

Parting Thoughts

Existing ideas shape programming

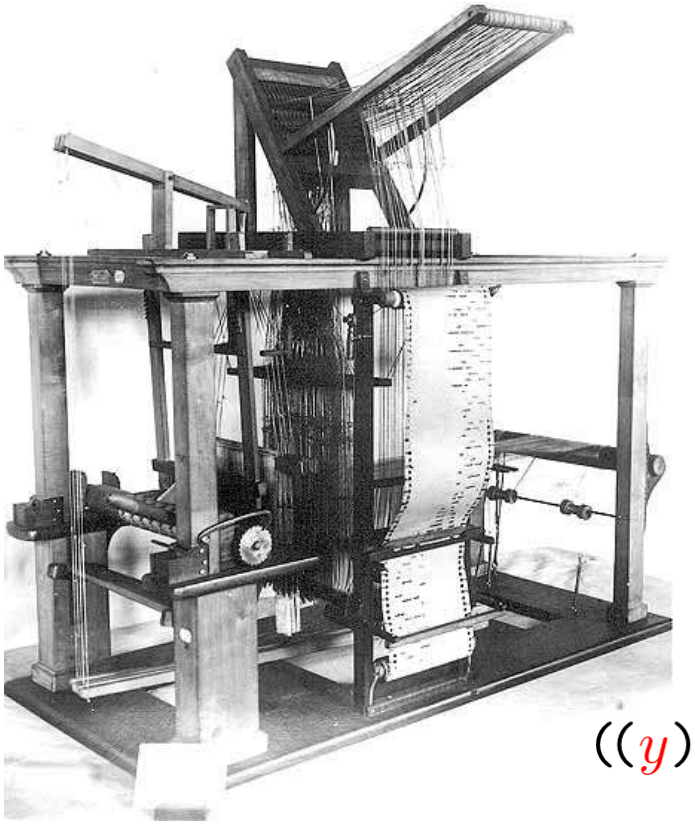


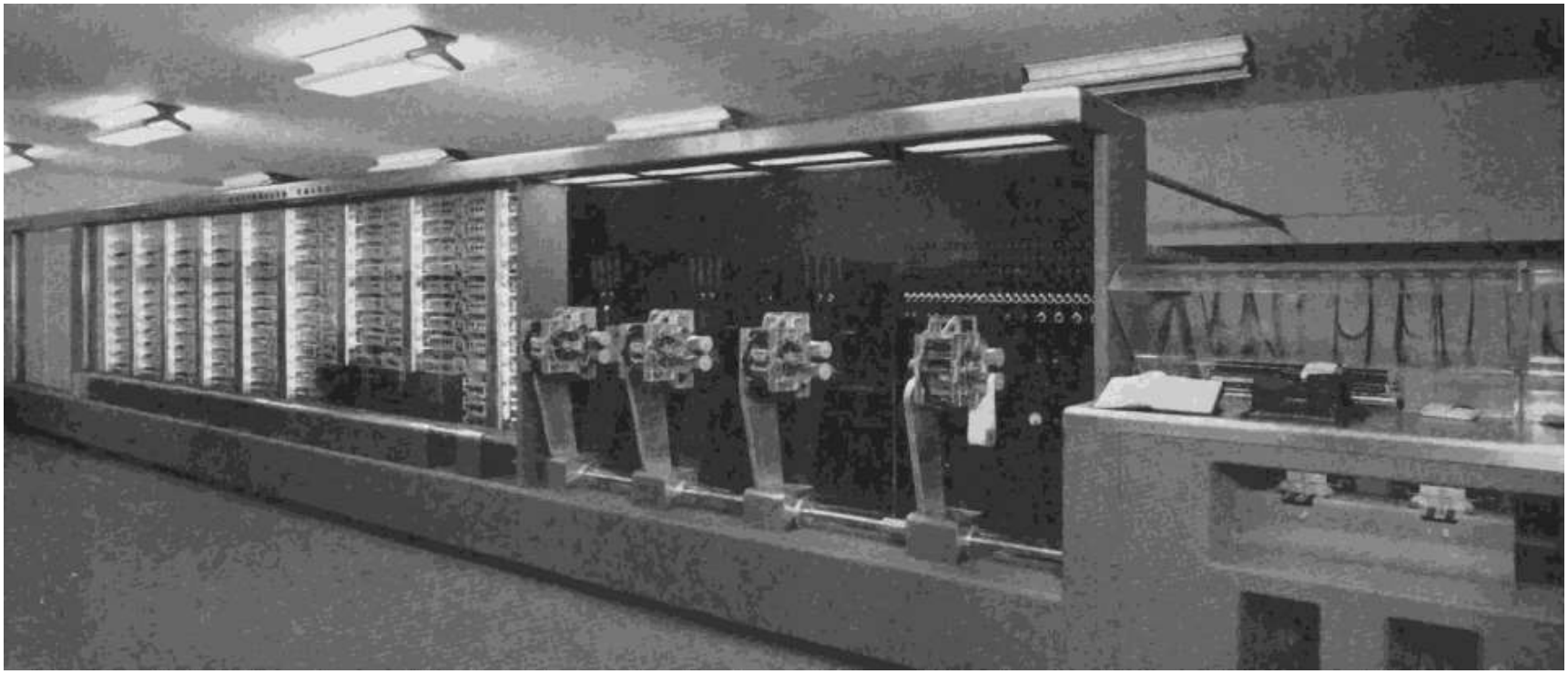
TABLE 1-1 Basic Identities of Boolean Algebra

(1) $x + 0 = x$	(2) $x \cdot 0 = 0$
(3) $x + 1 = 1$	(4) $x \cdot 1 = x$
(5) $x + x = x$	(6) $x \cdot x = x$
(7) $x + x' = 1$	(8) $x \cdot x' = 0$
(9) $x + y = y + x$	(10) $xy = yx$
(11) $x + (y + z) = (x + y) + z$	(12) $x(yz) = (xy)z$
(13) $x(y + z) = xy + xz$	(14) $x + yx = (x + y)(x + z)$
(15) $(x + y)' = x'y'$	(16) $(xy)' = x' + y'$
(17) $(x')' = x$	

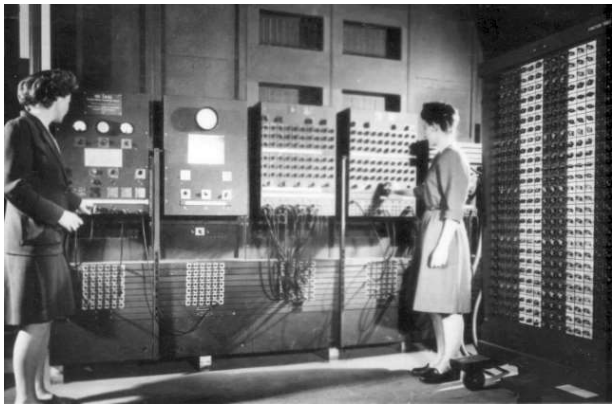
$$(\lambda y.\lambda x.y)\ z \quad \rightarrow \quad \lambda x.z$$

$$((y)\ \texttt{->}\ (x)\ \texttt{->}\ y)\ z \quad \rightarrow \quad (x)\ \texttt{->}\ z$$

Existing hardware shapes programming



Many women have shaped programming



You can shape programming