COMPUTING␣IN␣THE␣1960'S

Walter E. Brown, Ph.D.

< webrown.cpp @ gmail.com >

A little about me

- B.A. (math's); M.S., Ph.D. (computer science).
- Professional programmer for over 50 years, programming in C++ since 1982.
- Experienced in industry, academia, consulting, and research:
  - Founded a Computer Science Dept.; served as Professor and Dept. Head; taught and mentored at all levels.
  - Managed and mentored the programming staff for a reseller.
  - Lectured internationally as a software consultant and commercial trainer.
  - Retired from the Scientific Computing Division at Fermilab, specializing in C++ programming and in-house consulting.
- Not dead — still doing training & consulting. (Email me!)

Emeritus participant in C++ standardization

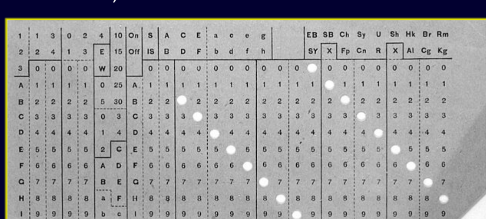
- Written ~175 papers for WG21, proposing such now-standard C++ library features as gcd/lcm, cbegin/cend, common_type, and void_t, as well as all of headers <random> and <ratio>.
- Influenced such core language features as *alias templates*, *contextual conversions*, and *variable templates*; recently worked on *requires-expressions*, operator<=>, and more!
- Conceived and served as Project Editor for *Int'l Standard on Mathematical Special Functions in C++* (ISO/IEC 29124), later incorporated into <cmath>.
- Be forewarned: Based on my training and experience, I hold some rather strong opinions about computer software and programming methodology — these opinions are not shared by all programmers, but they should be! ☺

What was it like almost 60 years ago?

- I started computing in those days, and would like to share with you my recollections of what it was like to be a programmer "back then."
- In this talk I will:
  - Show you some of the hardware we had to work with, …
  - Describe some of the system software that was available for programmers to use, and …
  - Offer some personal anecdotes dating from that era.
- Now, let's return to the days of punch cards, punched paper tape, unit record equipment, and disk drives the size of a laundry machine!

Hollerith's punch card, ~1885

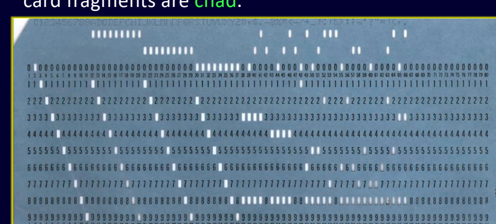
- 45 columns, with round holes and one corner cut:



- Sized (3¼ × 7⅜ in.) to match U.S. currency of that era and thus to take advantage of existing drawers, *etc.*

IBM's 80-column punch card was introduced in 1928

- Kept the 3¼ × 7⅜ in. dimensions of Hollerith's cards, …
- But now with (patented!) rectangular holes possible in 12 positions/column (10 until 1930); the punched-out card fragments are chad.



How important was the IBM punch card?

- "As late as the mid-1950s, punched card sales made up 20% of IBM's revenues and an astonishing 30% of its bottom line."
- "Remington Rand was IBM's main competitor in the punched card space. … The race of one-upmanship resulted in a slew of accounting developments focused on speed and automatic operations."
- "Until the early 1990s — long after IBM had ceased selling the punched cards for data processing — it was common practice for IBMers to use them for speaker notes for presentations, as they fit comfortably in the inside pocket of a suit jacket."

 8

IBM 026 printing keypunch machine (1949)

- Upper-case only.
- A typo was fatal; we had to discard the card.
- But we sometimes could save time by dup-ing the correct columns and rekeying only the wrong ones.

 9

A closer view of an IBM 026 keypunch

 10

The model 024 was the non-printing sibling of the 026

 11

Punch card media allowed some neat tricks

- Consider the following code, punched on a card:
  - `DEBUG = TRUE // ESLAF = GUBED`
  - (BCPL [1967] used slashes to introduce a comment.)
- Now imagine what happens to that card when you:
  - Remove it from the program deck, …
  - Rotate it 180° on its vertical axis, and …
  - Reinsert it into its original position in the deck.
- Now the same card reads:
  - `DEBUG = FALSE // EURT = GUBED`

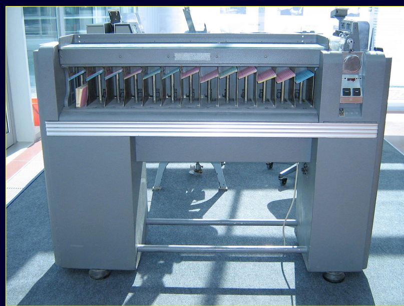
 12

Punch card media could also be frustrating [courtesy of "Peter"]

- "We were once debugging a program that would only sometimes fail. …
- "In th[o]se days, debugging … could take days. This problem did.
- "As the team was driven to despair, one guy in the debugging team started to watch the punched cards against the sunlight.
- "Turns out that there was one card which had a thin spot, and on dry days [*sic*!], the [card] reader would [sense] this [spot] as a hole.
- "Card replaced, problem solved."

 13

IBM 082 card sorter (1949) (hardware radix sort)



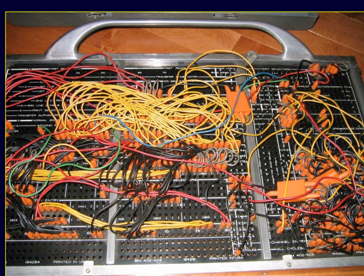
 14

Unit record equipment later became "programmable"

- "The earliest machines were hard wired for specific applications.
- "Control panels [also known as wiring panels or plugboards] were introduced [by Hollerith!] in 1906 ….
- "Removable control panels were introduced … in the 1920s.
  - "Applications then could be wired on separate control panels, and inserted … as needed.
  - "Removable control panels [were] used [when] the machines' … different applications required rewiring."

 15

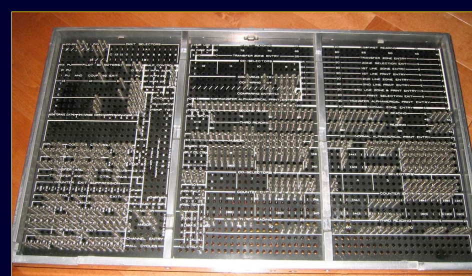
Plugboard from an IBM 402 Accounting Machine (front view)



- Dates from ~1948.
- Note the patch cords, plugged into hubs/jacks/sockets.
- Heavy! The handle is sized for two hands!

 16

Plugboard from an IBM 402 Accounting Machine (rear view)



 17

The "obstacle of the plugboard" — this was programming!

- "[T]he plugboard [was] a spaghetti-like array of cords and plugs that tell a machine what to do — move this information here, perform this action on that output and so forth.
- "These control panels … were nightmarishly con-structed Gordian knots of complexity and confusion.
- "They were slow, time-consuming, and above all, expensive."

— attributed to Francis Underwood
(who had worked w/ Arnold Spielberg, Steven's father)

 18

IBM 407 accounting machine (marketed 1949–1976, $800+/mo)

- Even so, plugboards were considered a huge advance.
- Previously, it was common for an entire machine to be broken apart in order to be reconfigured when-ever a new problem needed to be solved.

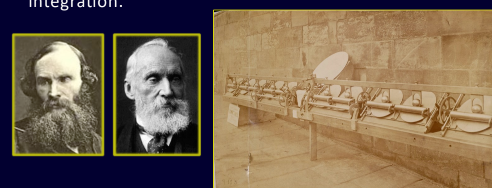
 19

Before plugboards: V. Bush's 1931 *differential analyzer*

 20

James (1822-1892) & William (Lord Kelvin, 1824-1907) Thomson

- Via descriptions in three papers, invented the differential analyzer (1876):
  - "A mechanical analogue computer designed to solve differential equations …,
  - "using wheel-and-disc mechanisms to perform the integration."



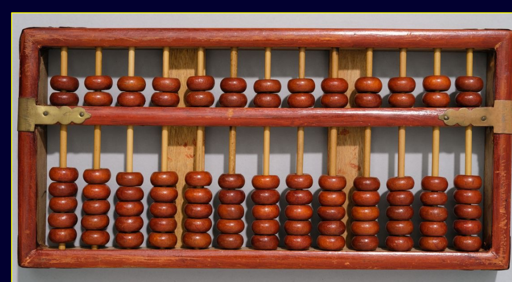
 21

The 1st computer I ever programmed …

- … was an IBM 650 (formally the Type 650 Magnetic Drum Data-Processing Machine), sold from 1953-1962:
  - A decimal machine, using bi-quinary encoding, with a 10-digit word length.
  - Main memory was a drum, 4" diameter, 16" long, rotating at 12,500 RPM, with a capacity (depending on the model) of either 10,000 or 20,000 digits.
- Rental was ~$3200/month (then the cost of a luxury car).
- I learned/coded (6 weeks) its machine language only:
  - Then a student, I never was allowed near the machine.
  - I really don't recall much else about it.

 22

BTW, bi-quinary encoding has been around for a looong time!



 23

An IBM 650 weighed > 5000 pounds (depending on configuration)

 24

Are they talking about me? (No!)

I don't know his exact age, but he speaks of having programmed an IBM 650 computer.

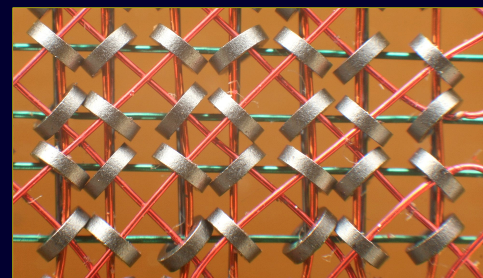
 25

Core memory came soon after

- Consisted of "Tiny donuts, made of magnetic material, [each] strung on [multiple] wires into an array":
  - Each toroid corresponded to a single bit, magnetized one direction for 0 and the other direction for 1.
  - Some wires were used to select the desired bit; other wires were used to detect and/or to reverse its direction.
- "[M]anufacturing [core memory] was a delicate job, …
  - "entrusted mostly to women using microscopes and steady hands …
  - "to thread thin wires through holes about the diameter of a pencil lead."

 26

Distance between these cores is ~1 mm



 27

Adoption of core memory

- Quickly became the dominant memory technology:
  - Far more reliable & longer-lasting than the vacuum tubes (*valves*) used by, *e.g.*, the ABC, ENIAC, and Colossus.
- Over ~20 years "The cost declined … from about $1 per bit to about 1 cent per bit":
  - Finally supplanted ~1975 by semiconductor technology.
  - In 1979, $5000 bought 128 kB (used) for a PDP-11/45 via a backplane, a power supply, and 9 circuit boards.
- But capturing a memory image (*e.g.*, of a failed program) is to this day still known as a *core dump*.

 28

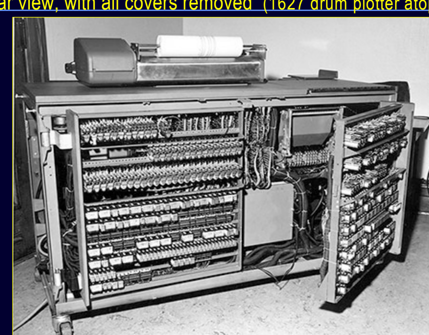
The IBM 1620 was a machine I knew extremely well

 29

Another view, with lower covers removed

 30

Rear view, with all covers removed (1627 drum plotter atop)



 31

Technology of the era: *discrete* components

- Later superseded by integrated circuit technology.
- First came SSI (small scale integration), then MSI, later LSI and VLSI.
- "Think of it! With VLSI we can pack 100 ENIACS in 1 sq. cm."

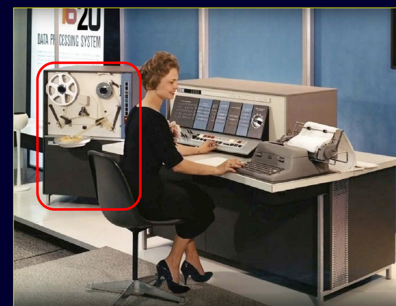
  — *Alan J. Perlis*
  (1st Turing Award recipient)

 32

A 1620 was desk-sized, with a built-in sprocket-feed typewriter

1621 Paper Tape Reader (150 char/sec)

1624 Paper Tape Punch (15 char/sec) also available



 33

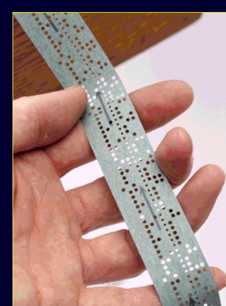
A closer view

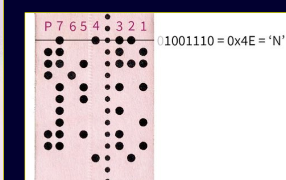
1621 Paper Tape Reader (150 cps)

1624 Paper Tape Punch (15 cps)

 34

Punched paper tape, whose 7-bit encoding gave rise to ASCII





 35

Other peripherals for this "small scientific market" machine

Note the absence of any disk!
… and of any printer!!

 36

Some of the 1620's architectural characteristics ①

- Smallest addressable memory unit was a 6-bit *digit*:
  - 4 bits (1, 2, 4, 8) used for *BCD* (*binary-coded decimal*), …
  - 1 bit used as a *flag* bit and 1 as a *check* bit (odd parity).
- A memory address consisted of 5 digits:
  - Allowed for 100,000 addressable digits, but no machine was sold with more than 60,000 — just too expensive.
  - The first 20,000 digits were housed in the CPU cabinet; the next 20,000 or 40,000 were in an expansion cabinet.
  - Memory cycle time was 20 μs/digit (model I).
- Cables, some as thick as a wrist, were a trip hazard, so were kept under the floor (raised for this purpose).

 37

Some of the 1620's architectural characteristics ②

- Most instructions were 12 digits in length:
  - A 2-digit op code, plus two 5-digit operands (P and Q).
  - Instructions had to be left-aligned on an even address.
- An operand was usually a memory address:
  - But, for the *immediate* op codes, Q was a 5-digit value.
  - Each right-flagged address denoted a level of indirection, a "Special Feature" (which meant it co$t extra).
- Instruction execution "generated a lot of … RF noise" that a nearby AM radio receiver could pick up:
  - So someone once wrote a program that generated specific freq's on demand and could thus produce music!

Some of the 1620's architectural characteristics ③

- A machine word varied in length (but always ≥ 2 digits):
  - Each word was addressed at the right (low-order digit), …
  - And continued to the left until a flagged digit was found.
  - A rightmost flagged digit denoted a negative number.
- A character was encoded in memory as a pair of adjacent digits:
  - This encoding predated ASCII, but correlated somewhat with the holes in a punch card.
  - *E.g.*, 41-49 represented A..I which all had a 12-punch; 51-59 represented J..R, which all had an 11-punch, *etc*.

Some of the 1620's architectural characteristics ④

- The Model I had not much circuitry for arithmetic:
  - The machine had been code-named CADET during its development.
  - The lack of native arithmetic led to a retronym for CADET: "Can't Add; Doesn't Even Try".
- Used a (patented!) table lookup to add/subtract/multiply:
  - Tables had to be loaded, starting at 00100, at each boot.
  - "Division is accomplished by a division subroutine or by the Automatic Divide special feature."
  - Hardware floating-point later became another special feature, emulated in software if absent — but no IEEE yet.

The 1620 console, made for program control & debugging

"Emergency Stop" switch

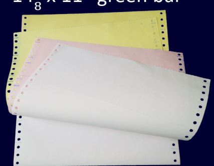
Memory Address Register & MAR Selector

Close-up of the MAR Display Selector

IBM 1620 in operation

IBM 1403 line printer came in several models (1959)

- Up to 1100 lines/min., 120|132 char's/line.
- Heavily soundproofed!
- Used continuous-form pin-fed paper, typically $14\frac{7}{8}$ x 11" green bar



 45

IBM 1403 in action

 46

IBM 1311 disk drive (size of a top-load clothes washer, 1962)

- Used an IBM 1316 removable *disk pack*:
  - 6 platters, 14" diam.
  - 10 recording surfaces.
  - Capacity of 2,000,000 digits/pack (1M char's).
- Up to 4 drives:
  - 1 master + 3 "slaved" drives.
  - 10 R/W heads ea.
  - Would "walk"!

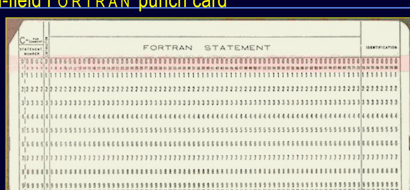
 47

IBM 1620 system software

- FORTRAN II
  - Compiler, loaded from punch cards (later from disk).
- GOTRAN
  - Fortran-like, featuring load-and-go operation.
- SPS (Symbolic Programming System)
  - Assembly language.
- MONITOR I
  - Single-user, disk-based operating system.
  - Time-sharing was still several years away.

 48

Fixed-field FORTRAN punch card



- Col 1: 'C' to introduce a comment, else blank
- Col's 2-5: optional statement number (*i.e.*, a label)
- Col 6: any non-blank character when a continuation
- Col's 7-72: Fortran (actually, FORTRAN) statement
- Col's 73-80: optional card sequence number

 49

FORTRAN II idiosyncrasies ①

```
  I = 1
  J = 2
  PRINT 9, I, J
  CALL SWAP( 1, 2 )
  I = 1
  J = 2
  PRINT 9, I, J
  STOP
9 FORMAT( 2I3 )
  END
```

```
1  2
2  1
```

 50

FORTRAN II idiosyncrasies ②

- This code starts a loop of 100 iterations via counter `I`:
  - `DO 10 I = 1,100`
  - The loop's body follows on subsequent lines, up to and including the statement labelled `10`.
- In contrast, this near-identical code is an assignment:
  - `DO 10 I = 1.100`
  - It updates a variable `DO10I` to the value `1.1`.
  - Why? Because spaces are not significant, hence ignored!
  - Also, variable `DO10I` has type `REAL`, because undeclared variables are implicitly typed based on their initial letter: `I`..`N` → `INTEGER`, otherwise `REAL`. ("God is real, unless declared integer.")

FORTRAN II idiosyncrasies ③

- There were:
  - No relational operators (<, >, *etc.*), …
  - No logical operators (not, and, or), and …
  - Neither truth values (true, false) nor a boolean type.
- Instead, we had the arithmetic IF statement, *e.g.*:
  - `IF( X – Y ) 10, 20, 30`
  - Evaluates the expression, then branches per its *signum*
    - to the statement labelled `10` if negative (*i.e.*, X < Y),
    - to the statement labelled `20` if zero (*i.e.*, X==Y), or
    - to the statement labelled `30` if positive (*i.e.*, X > Y).

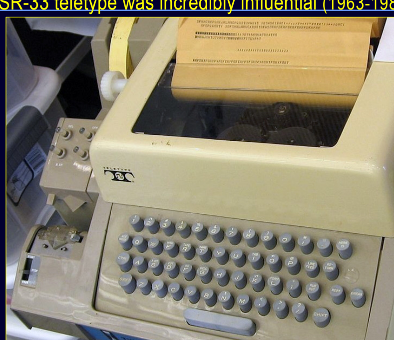
FORTRAN II idiosyncrasies ④

- Names were limited to 6-character length. (Led to such naming puns as `EVALU8`, *etc.*)
- This length restriction originated with 6-bit character encodings, packed, on a 36-bit word machine:
  - "36-bit computers were popular in the early mainframe computer era from the 1950s through the early 1970s."
  - 36 bits is enough to encode 10-digit integers.
  - Such machines included IBM's 701/704/709/7090/7094, UNIVAC's 1103/1103A/1105, GE's 600, Honeywell's 6000, and Digital's PDP-6/PDP-10.
  - Character encodings included 6-bit BCD, 6-bit ASCII (no lower case), Multics' 9-bit characters, *etc.*

An anecdote

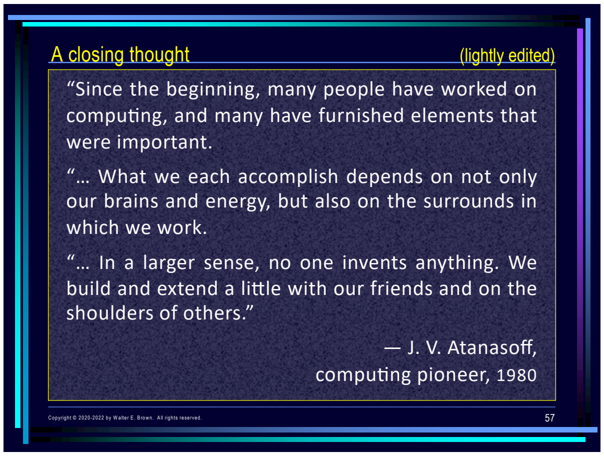
- In FORTRAN II (and later), an array was declared via a `DIMENSION` statement, *e.g.*, `DIMENSION A(10)` :
  - However, that keyword was often misspelled (typically by novice programmers) as `DIMENTION`.
  - Yet, on the 1620, that misspelling compiled successfully!
- Experimentation revealed that (to conserve code size?) the parser treated any long-enough token starting with a `D` as introducing a `DIMENSION` statement:
  - A local consensus was quickly reached, and …
  - For a time, local student programs would routinely declare their arrays via a `DUMBWAITER` statement!

The ASR-33 teletype was incredibly influential (1963-1981)

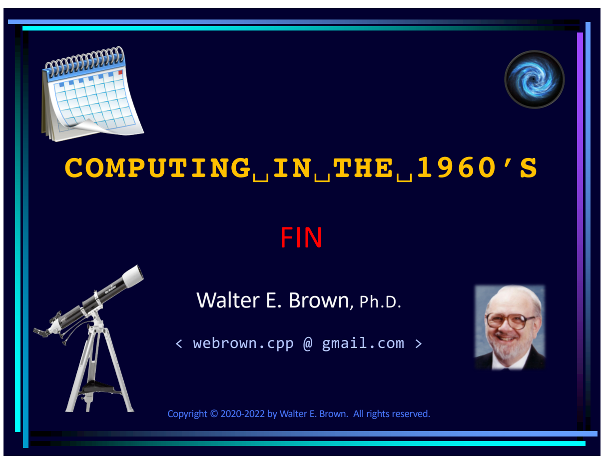


A few of the ASR-33's achievements

- First device to use ASCII, which standard it influenced.
- "BASIC was designed to be written and edited on a low-speed Teletype Model 33."
- Used "in the pilot program for computer assisted instruction" at Stanford University.
- "In 1971, Ray Tomlinson chose the @ symbol on his Teletype Model 33 ASR keyboard for use in network email addresses."
- Unix™ (and Linux, *etc.*) serial ports' names are /dev/tty, which abbreviates "teletype."
- Over 600,000 were manufactured.


A closing thought
(lightly edited)
"Since the beginning, many people have worked on computing, and many have furnished elements that were important.
"… What we each accomplish depends on not only our brains and energy, but also on the surrounds in which we work.
"… In a larger sense, no one invents anything. We build and extend a little with our friends and on the shoulders of others."
— J. V. Atanasoff,
computing pioneer, 1980
Copyright © 2020-2022 by Walter E. Brown. All rights reserved.
57


J. V. Atanasoff being hooded, 1981
MORAVIAN
Copyright © 2020-2022 by Walter E. Brown. All rights reserved.
58


COMPUTING␣IN␣THE␣1960'S
FIN
Walter E. Brown, Ph.D.
< webrown.cpp @ gmail.com >
Copyright © 2020-2022 by Walter E. Brown. All rights reserved.