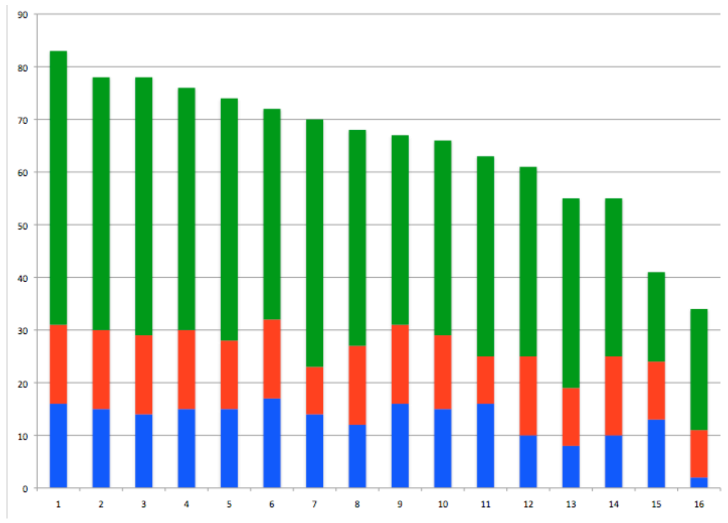
COS 109 Midterm Exam, Fall 2024

I graded this myself. The median is 67.5 this year, compared to 62 last year, so perhaps the exam was easier, or perhaps it's all kind of random. As always, there were evident problems with how to represent information in bits and bytes and how many of them are needed. There seemed to be a fair amount of confusion about how to state an algorithm clearly.

The three values in each column below are for parts 1, 2 and 3, reading from the bottom.



1. (20 points, 2 each) Short Answers. Circle the right answer or write it in the space.

(a) A 7-digit phone number like 5551212 requires 7 bytes if stored as ASCII characters. How many bytes are required to store any particular such phone number if it is stored as a binary number?

3 bytes. A 7-digit number is between 0 and 10^7, so 24 bits is enough. A surprising number of people came up with 4 bytes.

(b) If I were to View the Source for a typical modern web page, which of these would be ***unlikely*** to occur there? Circle all that are ***not likely*** to be present.

CSS **C++** **HTML** **Java** **JavaScript** **Python**

C++, Java, Python. All are programming languages that have nothing to do with web pages.

(c) In Unicode, the first character of the Phoenician alphabet is hex `10900` and the last is hex `1091F`. How many Unicode characters are there in Phoenician?

32. The range is 00 to 1F inclusive, which is 32.

(d) What is the minimum number of bytes required to store an arbitrary Phoenician character in the Unicode encoding?

3 bytes. The values in the question are 3 bytes long.

(e) I'm thinking of a *floating-point* number between 1 and 1,000. About how many guesses would it take you to figure it out to within 3 decimal places?

20 guesses. There are a million possible values. Missed by more than I had expected.

(f) If **i**, **j** and **k** are integers, how many 1 bits (i.e., bits that have the value 1) are there in the binary representation of the number $\mathbf{2^i \times 2^j / 2^k}$?

1 **2** **3** **8** **(i + j) / k** **i x j / k** **none of these**

1. It's a power of 2 regardless of the values of i, j and k. Mostly well done.

(g) If a 5-inch integrated circuit wafer like the ones passed around in class has 100 chips, about how many of the same chips would there be on a 15-inch wafer?

900. The area grows with the square of the diameter. Almost everyone got this freebie.

(h) Suppose a RAM package (like those passed around in class, and analogous to the one pictured here) has 8 chips on each side and each chip contains 128 gigabits. What is the total capacity of the package in gigabytes?

256 GB. 8 chips x 2 sides x 128 Gbits per chip / 8 bits per byte.

(i) A recent Slashdot story says "Modern processors support about 280 TB of addressable memory." How many bytes minimum would it take to represent an address (that is, a location in RAM) in such a processor?

1 **2** **3** **4** **6** **8** **16** **256** **280** **none of these**

6. "About 280" means that it was really 256, a power of two.

(j) A _____**compiler**________________ is a program that translates a statement like "c = 2 * pi * r" into a sequence of ______**assembly language or assembler**_____ instructions like "load 2; mul pi; mul r; store c". What words belong in the blanks?

Not "machine", which is just bits.

2. (15 points) Playing with Toys

Suppose that the Toy machine is augmented with a new instruction `CHSIGN` that changes the sign of the value in the accumulator. That is, if the accumulator value is positive it becomes negative, and if it is negative it becomes positive; zero is unchanged. Here is a small program that uses the `CHSIGN` instruction, with reminders about what the instructions do:

```
MORE   GET              get a number from user, place it in accumulator
       IFZERO END       if accumulator value is 0, go to END
       IFPOS  MORE      if accumulator value is >= 0, go to MORE
       CHSIGN           change sign of value in accumulator
       PRINT            print value in accumulator
       IFPOS  MORE
END    STOP
```

(a) If you run this program and give it the sequence of input numbers **2 1 –7 3 –8 –4 –6 5 9 0**, what number or numbers does it print, if any?

7 8 4 6. It prints the positive value of input numbers that are negative.

(b) How does the running time of this program vary in proportion to **n**, the number of input numbers?

log n **n** **n log n** $\mathbf{n^2}$ $\mathbf{2^n}$ **no way to tell**

n. Simple linear algorithm: do essentially the same thing to each input number

(c) In his 1950 paper *Computing Machinery and Intelligence*, Alan Turing includes an informal description of a computer rather like our Toy machine. He says "We wish to be able to arrange that the sequence of orders can divide at various points, continuing in different ways according to the outcome of the calculations to date."

Name one Toy instruction that achieves this effect.

IFZERO, IFPOS.

(d) Turing also said "It is much easier to work in the scale of __________ than any other, because it is so easy to produce mechanisms which have __________ positions of stability." What number goes in the blanks?

2.

(e) The computer described in Von Neumann's 1946 paper includes an instruction that shifts the bits in the accumulator ***to the left*** by one position, replacing the vacated position on the right end by a 0 bit. What arithmetic operation does this shift perform on the binary number in the accumulator?

multiplies by 2. I really wanted to see what it was multiplying by.

(f) If the instruction instead shifts the bits in the accumulator ***to the right*** by one position, discarding the rightmost bit and replacing the vacated position on the left end by a 0 bit, what arithmetic operation does this shift perform on the number in the accumulator?

divides by 2. Same

3. (55 points, 5 each) Miscellaneous

(a) Consider the two hexadecimal numbers **`FOO`** and **`OFF`**.

(i) What is the sum **`FOO + OFF`** expressed in hexadecimal?

FFF. Nearly everyone got this freebie.

(ii) What is that sum expressed in decimal?

4095. No need to do arithmetic if you know the powers of two; it's 2^12-1.

(iii) What is the value of **`FOO + OFF + 1`** expressed in hexadecimal?

1000. Again, no arithmetic needed. Look for familiar patterns of powers of 2.

(b) Apple's M3 chips come in three sizes, all made with the same 3 nm technology. Suppose that the base M3 is 1 cm x 1 cm, the M3 Pro is 1.5 cm x 1.5 cm and the M3 Max is 2 cm x 2 cm. (These numbers are purely speculative.)

(i) If the base M3 chip has 10^{10} transistors, approximately how many is the M3 Pro likely to have?

2.25 * 10^10. A gift.

(ii) Approximately how many transistors is the M3 Max likely to have?

4 * 10^10. Another gift!

(c) Suppose that the CS department wants to list all students who are enrolled in both COS 217 and COS 226 this semester. Assume that there are **N** students in each course.

(i) Describe an ***efficient*** algorithm for listing all students who are in both courses. Your algorithm should be ***as efficient as possible***, not something easily improved upon. Your description should be brief (10 to 15 words is enough) but clear about the basic approach or idea.

Sort the two lists of names together, then go through them looking for adjacent duplicates. Or sort one list, use binary search from the other list. Either way, it's n log n for the sorting if you use an efficient algorithm.

Almost no one said anything like this. Much arm-waving about "then search" but not a hint of how that search would be done.

(ii) How does the running time of your algorithm depend on **N**, the number of students in each course?

n log n

(d) In November 2022, the SI units were updated by adding names for two new big numbers: peta, exa, zetta and yotta are now followed by **ronna** and **quetta**.

(i) What power of 2 is closest to the power of 10 that quetta represents?

2^100. Just count up from familiar ones.

(ii) How many petabytes are there in a quettabyte?

10^15. Same thing.

(e) Suppose we filled Friend 008 with disk drives like the ones that were passed around in class and shown in the picture here. If each drive holds 1 TB, estimate *very roughly* the number of exabytes that the room would hold. Ignore power, wires, desks, seats, etc., and assume that the whole volume of the room is filled.

Maybe 2 EB? There were some disturbingly large disks. Typical small ones were passed around in class and are shown in the book. Room estimates were pretty decent, though some were too big. There were a handful of two-dimentional computations. It did say that we were filling the room, not tiling the floor. Minor penalty for excess precision.

(f) In *Click Here to Kill Everybody*, Bruce Schneier says "If 100 systems are interacting with each other, that's about 5,000 interactions and 5,000 potential vulnerabilities resulting from those interactions. If 300 are all interacting with each other, that's 45,000 interactions."

(i) If 1,000 systems interact with each other, approximately many interactions would Schneier report?

500,000. Each of 1,000 interacts with 999 others, then divide by two since A interacting with B is the same as B interacting with A.

(ii) How does the number of such interactions grow as a function of or in proportion to the number of systems?

N^2. As discussed at length in class, practically the definition of quadratic.

(g) The book *The Zero Marginal Cost Society* says that in 2007 there were 10 million sensors connected to the Inernet, and that there will be 10 trillion sensors in 2027.

(i) If this growth is a smooth exponential process, how many years would it take for the number of sensors to double?

1 year. The number goes up by a factor of 1,000 in ten years.

(ii) What is the rate of growth *per month* of the number of sensors?

6%. Rule of 72. You can use a calculator but given that all of the numbers involved are very rough approximations, giving 3 "significant" figures isn't appropriate.

(h) Quickies (1 or 2 word answers):

Your prox card uses a discovery made by what Princeton professor around 1830? __**Joseph Henry**_______

What major software system makes a brief appearance in *Jurassic Park*? ____ **Unix** _______

"Coding pioneer ______ Lovelace" (*NYTimes* crossword clue, 9/21/24. 3 letters) ____ **Ada** _________

Leibniz and Babbage corresponded about the use of binary numbers: true or false? ___ **false** ________

Moore's Law is named for an anti-trust case involving Gordon Moore: true or false? _____ **false** _____

Most people got all of these easy ones.

(i) Here are five hexadecimal numbers corresponding to the traditional colors Princeton orange, Harvard crimson, Yale blue, Dartmouth green and Army gray. Write the proper school name beside each color.

`00A93E`	`Dartmouth`
`0F4DA2`	`Yale`
`B2B4B3`	`Army`
`C90016`	`Harvard`
`E77500`	`Princeton`

Mostly well done.

(j) "One if by land, and two if by sea." Suppose that some modern-day Paul Revere wants to send more extensive information about an invading force. He wants to encode these three items in *as few total bits as possible*:

- whether the force is coming by land, sea or air
- the approximate size of the force to the nearest 1,000, with a maximum of 15,000
- what time of day or night the force set off, to the nearest hour.

How many total bits does Paul need to use, and why?

11 = 2 + 4 + 5. Some confusion in evidence, but more often carelessness, e.g., missing the word "nearest".

(k) Morse code uses combinations of one to five dots and/or dashes to represent letters, digits, and punctuation marks. For example, `E` is a single dot (•), `A` is dot-dash (•–), and `Q` is dash-dash-dot-dash (––•–). Suppose you are designing a new version of a Morse-like code, in which every character will consist of some pattern of *exactly* 6 dots and/or dashes. Describe briefly how you would systematically assign upper case letters and digits to patterns of 6 dots and dashes. Write down enough of your patterns or explain them so clearly that there is no ambiguity about your design.

Not well done. It was sufficient to say "number the letters and digits in binary starting at 0 (or 1)." We did talk about how to do this in class a couple of times. When you start shifting patterns of dots and dashes, it's easy to make a mistake, or to waffle too much about "and so on." Also, read the question carefully; it says "upper case letters".