COS 109 Midterm Exam, Fall 2017  In class, Thursday, October 26

PRINT your name here ____________________________________________

Do not discuss the exam with, or accept help from, anyone.  You must write and sign this statement:

“This examination represents my own work in accordance with University regulations.”

Rules

This examination is open-book and open-note:

• You may use the textbook, course notes, your own notes, corrected problem sets and solutions, old exams and answer sheets from the course web page, lab instructions, etc.
• You may use a calculator.
• You may not use anything else; specifically, you may not use a computer, phone or tablet (except that you can use the calculator program on one of these, and you can use your computer to view course notes if you did not print them).  You cannot use your computer to access the internet except for the course materials.

Procedure

There are a total of 90 points for the questions; use the point values for each question to allocate your time appropriately (one point per minute).

Write your answers directly on these pages; if you need more space, there is a blank page at the end and you can write on the backs of pages. It's quite all right to be brief as long as you're clear. We have tried to leave plenty of room for answers; if you are writing or computing a lot, you may be off on the wrong track.

Good luck.
1. (15 points)

(a) State Moore’s Law

Moore’s Law states that the thickness of lines in an integrated circuit get thinner over time. As a result, the number of transistors that can fit into a fixed area (at a fixed cost) doubles every 18 months.

(b) In 1991, a disk that had 1 gigabyte of memory cost $1000. Now, a disk that holds 2TB of information costs $80. Have disk prices and capacities changed faster or slower than Moore’s Law?

The cost has gone down by a factor of $12.5 = 1000/80$. The capacity has gone up by a factor of $2000 = 2\text{TB}/1\text{GB}$. Multiplying these numbers means that the cost per byte has gone down by a factor of $25,000$. 25,000 is between $2^{14}$ and $2^{15}$ which corresponds to between 14 and 15 doublings. Moore’s Law suggests that this would happen over a period of between 21 and 22.5 years. So, this is slightly less than the 26 years between 1991 and 2017. So, the prices/capacity has changed slightly slower than Moore’s Law would predict.

2.(50 points, 5 each) Short Answers. Write your answer in the space provided.

(a) What is the difference between ASCII and Unicode in terms of space (in bits and bytes) used to store a character and the number of characters that can be represented?

ASCII requires 1 byte and can represent 256 characters. Unicode requires 2 bytes and can store more than 65,536 characters.

(b) What is Memex and who proposed it?

Memex was an idea proposed by Vannevar Bush. The idea was that Memex would record all information. Although Memex is not precisely what Google does, it was a good predictor from many years earlier.

(c) I have set the color for my screen background to ABCDEF; what are the components of this color in decimal numbers?

\[
\begin{align*}
AB &= 10 \times 16 + 11 = 171 \\
CD &= 12 \times 16 + 13 = 205 \\
EF &= 14 \times 16 + 15 = 239 \\
\text{So, red is 171, green is 205, blue is 239}
\end{align*}
\]
(d) What is distinctive about the von Neumann architecture? What was the first computer to employ this architecture?

The von Neumann architecture is an architecture in which program instructions and data are stored in the same memory. The advantage of this is that programs can be modified as it executes. The disadvantage is that instructions can be read as data and vice versa.

The first machine with this architecture was the Johniac which was built at IAS

(e) There are about 6000 tweets sent every second. If we wanted to give every tweet sent in a year a unique ID, how many bits would it take for this representation?

\[ 6000 \text{ tweets per second} \times 60 \text{ seconds/minute} \times 60 \text{ minutes/hour} \times 24 \text{ hours/day} \times 365 \text{ days/year} \]

amount to about 189 billion or 189 x 10^9 tweets per year. 10^9 is about 2^{30} and 189 is a bit less than 2^{8}. So, 38 bits or 5 bytes would suffice.

(f) Give an example of a problem that requires exponential time for its solution.

The Towers of Hanoi problem requires exponential time for its solution. Every time the input size increases by 1, the solution time doubles.

(g) As the number of transistors in a CPU has grown, CPUs have been able to add functionality that did not previously fit into the CPU. Give an example of this added functionality.

CPUs added caches which meant that there was memory in the CPU which would allow recently used information to be recalled more quickly. Next, CPUs became multi-core which meant that there could be more processors on the CPU. After that, CPUs added graphics capabilities which meant that separate graphics cards were no longer needed.

(h) What are the differences between assembly language and machine language?

Assembly language is written with mnemonics, so words like ADD, STORE, LOAD, … in order to be more human readable. Machine language is written in terms of 1’s and 0’s so as to be closer to the machine. For example, LOAD might be rewritten as the instruction 11001010

(i) How many kilobytes in an exabyte?

An exabyte is a thousand petabytes
A petabyte is a thousand terabytes
A terabyte is a thousand gigabytes
A gigabyte is a thousand megabytes
A megabyte is a thousand kilobytes
So, there are 10^{15} kilobytes in an exabyte. This is a quadrillion.
(j) Give the truth table for the function \( D = ((\neg (A \lor B)) \lor (A \land B)) \lor C \)

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>NOT (A OR B)</th>
<th>A AND B</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
3. (25 points) Machines

Here is a program in the Toy assembly language, with reminders about what the instructions do.

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET</td>
<td>get a number from keyboard into accumulator</td>
</tr>
<tr>
<td>IFZeros</td>
<td>if the value is 0, go to Done</td>
</tr>
<tr>
<td>IFPOS</td>
<td>if accumulator is greater than zero, go to Bar</td>
</tr>
<tr>
<td>Foo</td>
<td>ADD 1 to accumulator</td>
</tr>
<tr>
<td>Bar</td>
<td>SUB 1 subtract 1 from the accumulator</td>
</tr>
<tr>
<td></td>
<td>PRINT print contents of accumulator</td>
</tr>
<tr>
<td>Done</td>
<td>STOP</td>
</tr>
<tr>
<td></td>
<td>GOTO Foo go to instruction labeled Foo</td>
</tr>
<tr>
<td></td>
<td>GOTO Bar go to instruction labeled Bar</td>
</tr>
</tbody>
</table>

What does the program do? In your description describe what is printed as the program executes on various inputs.

It reads an input. If the input is zero, it stops. If the input is positive, it prints out the numbers counting down from that number down to 0. If the number is negative, it prints out the numbers counting up from that number to 0.