# CMPSCI 105 Midterm Exam Solution
# Spring 2010
# March 10, 2010
# Professor William T. Verts

## *GENERAL KNOWLEDGE*

<1> 16 Points – (1 point each) – Fill in your answer into the box at the left side of each question. Show your work on the back of a page if you want us to consider partial credit.

| | |
|---|---|
| **FALSE** | 1. True or False: A 3½-inch diskette is a "hard disk". |
| **FALSE** | 2. True or False: A USB flash drive is a "hard disk". |
| **FALSE** | 3. True or False: The binary value of the number π (pi) ***can*** be stored into computer memory with its full precision.<br>Pi is irrational, which means that its fraction goes on forever without repeating. You cannot put an infinite number of digits into a finite number of computer bits. This also holds for rational numbers that go on forever, such as ⅓ = 0.333333… (only rational numbers with binary representations that need no more bits than the given precision will be correctly stored). |
| ~~**TRUE**~~<br>**FALSE** | 4. True or False: With one byte you can represent all ***signed decimal*** values between 0 and +255, but no other values.<br>The question was graded as **TRUE**, but actually this is **FALSE**. When I made up the grading key I misread the question as saying "unsigned" instead of "signed". In unsigned arithmetic one byte can represent all values between 0 and 255, but in signed arithmetic one byte can hold values only in the range from -128 to +127, so the proper answer is false. Anyone who answered false and had it marked wrong can get this point back. |
| **$2^9$-1 = 511** | 5. What is the largest ***unsigned*** number that will fit into 9 bits?<br>With 9 bits there are $2^9$=512 distinct patterns. One of those patterns is zero, so the largest is $2^9$-1 or 511 (either answer is acceptable). |
| **1110110** | 6. Convert the decimal (base 10) number 118 into ***binary*** (base 2).<br>118÷2 = 59R**0** Least significant bit<br>59÷2 = 29R**1**<br>29÷2 = 14R**1**<br>14÷2 = 7R**0**<br>7÷2 = 3R**1**<br>3÷2 = 1R**1**<br>1÷2 = 0R**1** Most significant bit |

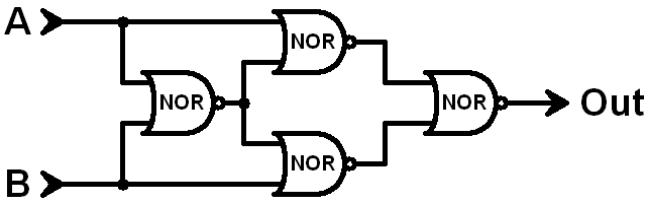
| | |
|---|---|
| **76** | 7. Convert the decimal (base 10) number 118 into ***hexadecimal*** (base 16).<br>118÷16 = 7R**6** Least significant digit<br>7÷16 = 0R**7** Most significant digit<br>Clever students will recognize that the binary answer to problem #6 can be partitioned into 0111-0110 and each set of four bits can be converted into its equivalent hexadecimal digit to get the same answer. |
| **9D** | 8. Convert the binary (base 2) number $10011101_2$ into ***hexadecimal*** (base 16) using the ***bit-partitioning*** method.<br>The number is partitioned into 1001-1101 and each 4-bit block is converted separately:<br>$1001 = 1\times2^3 + 0\times2^2 + 0\times2^1 + 1\times2^0 = 8 + 0 + 0 + 1 =$ **9**<br>$1101 = 1\times2^3 + 1\times2^2 + 0\times2^1 + 1\times2^0 = 8 + 4 + 0 + 1 = 13 =$ **D** |
| **235** | 9. Convert the binary (base 2) number $10011101_2$ into ***octal*** (base 8) using the ***bit-partitioning*** method.<br>This uses the same process as in problem #8, except that there are three bits in each partition. The partitioned number is therefore 010-011-101 (the left-most packet requires an additional leading zero):<br>$010 = 0\times2^2 + 1\times2^1 + 0\times2^0 = 0 + 2 + 0 = 2$<br>$011 = 0\times2^2 + 1\times2^1 + 1\times2^0 = 0 + 2 + 1 = 3$<br>$101 = 1\times2^2 + 0\times2^1 + 1\times2^0 = 1 + 0 + 1 = 5$ |
| **100001100** | 10. What is the ***sum in binary*** (base 2) of the two binary numbers $10011101_2$ and $1101111_2$?<br>Here are the numbers lined up, with dots to indicate the carries (there is a carry out of every digit):<br>`  .......`<br>`  10011101`<br>`+ 1101111`<br>`100001100` |
| **12-2*i*** | 11. What is the ***sum*** of complex numbers **9-6*i*** and **3+4*i*** ?<br>$(9+3) + (-6+4)i = 12-2i$ |
| **51+18*i*** | 12. What is the ***product*** of complex numbers **9-6*i*** and **3+4*i*** ?<br>Using the "FOIL" method:<br>$(9\times3) + (9\times4i) + (-6i\times3) + (-6i\times4i) = 27 + 36i - 18i - 24i^2$<br>But $i^2 = -1$ so $-24i^2 = +24$. This becomes $(27+24) + (36-18)i = 51+18i$. |
| **20200** | 13. When counting in base 3, what is the next number after $20122_3$?<br>The sum 20122+1 causes the rightmost digit to go from 2 to 0 and generate a carry to the second digit, which then itself goes from 2 to 0 and generates a carry to the third digit, which goes from 1 to 2. There is no carry out of the third digit, so the process stops at this point. |

| | |
|---|---|
| $\lceil 4.713\ldots \rceil = 5$ | 14. If $\log_5(1969) = 4.7130001226125027565\ldots$, how many base 5 digits are required to represent the decimal number 1969?<br>The actual base-5 number equivalent to 1969 is $30334_5$ which contains 5 digits. Because the leftmost of those digits is 3, on the scale of base-5 digits 0…4, not all of that fifth digit is completely filled up. This is what the logarithm is telling us by giving an answer of 4.713…. It says that 4 complete base-5 digits are needed but only .713… of the fifth base-5 digit is required. The number of digits in any number is an integer (you can't write down a fraction of a digit position), so taking the ceiling of that number (the ⌈ and ⌉ symbols) gives us the smallest integer greater than or equal to 4.713…, which is 5. |
| `550` | 15. In the UNIX command `chmod _____ filename` what must be in the blank to set permissions to **`r-xr-x---`** ?<br>In this problem you do not know the current value of the permissions, but you do know what each one must be. This requires the absolute form of the `chmod` command. The permissions mask `r-xr-x---` has the binary equiavalence `101101000`, which is partitioned into `101-101-000` for conversion to octal (base 8):<br>$101 = 1\times2^2 + 0\times2^1 + 1\times2^0 = 4 + 0 + 1 = 5$<br>$101 = 1\times2^2 + 0\times2^1 + 1\times2^0 = 4 + 0 + 1 = 5$<br>$000 = 0\times2^2 + 0\times2^1 + 0\times2^0 = 0 + 0 + 0 = 0$ |
| `u+rw` | 16. In the UNIX command `chmod _____ filename` what must be in the blank to add read & write permission to the user?<br>In this problem we want to change only certain permissions, but we neither know what the permissions are currently nor what they'll be afterwards. This requires the symbolic form of the `chmod` command. The user is `u`, adding permissions is indicated by `+`, read permission is `r`, and write permission is `w`, so the permissions mask will be `u+rw`. |

<2> 4 Points – Fill in the boxes below to show the expansion of the base 3 number 2011:

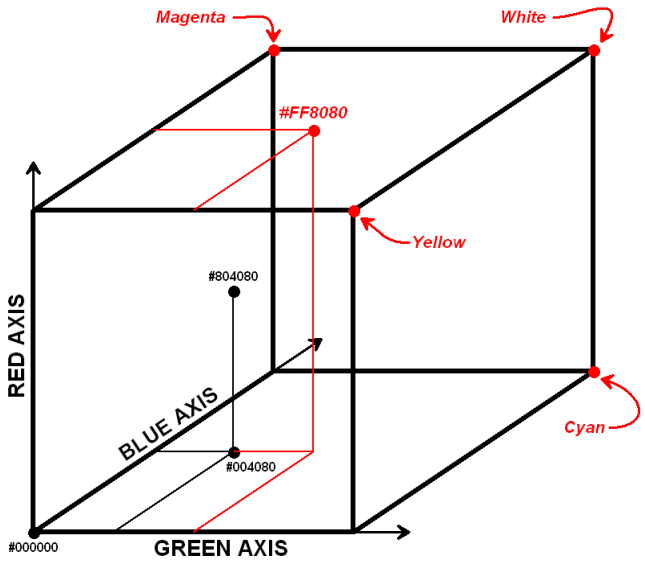
$$= \boxed{2}\times\boxed{3}^{\boxed{3}} + \boxed{0}\times\boxed{3}^{\boxed{2}} + \boxed{1}\times\boxed{3}^{\boxed{1}} + \boxed{1}\times\boxed{3}^{\boxed{0}}$$

<3> 8 Points – Trace the following gate circuit and show its output for all combinations of input values.




| In | | Output |
|---|---|---|
| A | B | |
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

<4> 5 Points – The RGB color cube below shows the positions of the colors **#000000**, **#004080**, and **#804080**. Show the locations of Magenta, Cyan, Yellow, White, and **#FF8080**.




The positions of the saturated colors (Magenta, Cyan, Yellow, and White) are all on corners of the cube. For example, Magenta is where it is because it has maximum red and maximum blue, but no green. Cyan, on the other hand, has maximum blue and maximum green, but no red. White has all three values maximized, so it will be on the corner farthest away from **#000000** (Black). These match the diagram in the book. The tricky one is the **#FF8080** color. The **FF** indicates that the color has maximum red, so it'll be somewhere on the top face of the cube. Both the green and blue values are **80**, which is halfway between **00** and **FF**, so the green and blue positions will both be halfway along their respective axes. Thus, the position of **#FF8080** is in the dead center of the top face of the cube. The positions of the colors **#804080** and **#004080** were given for reference.

## *UNIX, FTP, and Web Design*

<5> 4 Points – (1 point each):

1. In the Internet address **`garbo.uwasa.fi`**, what does the **`.fi`** stand for?

   **Finland**

2. In the Web address **`http://www.cs.umass.edu/~verts/`**, what is the name of the actual file that is fetched by the browser?

   **`index.html`** or **`index.htm`** (because no file name is specified in the URL, the browser will fetch the `index.html` or `index.htm` file by default.)

3. Where on my account is the file which was fetched in part 2 actually located (i.e., in which folder on my account is the file stored)?

   **`public_html`** (even though the `public_html` folder is not explicitly part of the URL, it is implicitly assumed – remember where Web files were stored on your UNIX account in lab #2?)

4. Does it matter to a modern browser (running on a Mac or on a recent version of Windows) if a Web document has **`.htm`** or **`.html`** as its file extension?

   **No**. (Both are treated the same way. Historically, the `.htm` extension is present at all only because PCs at the time the Web started to become popular were running MS-DOS or Windows 3.1, which allowed no more than three characters in the file name. It wasn't until Windows 95 in 1995 that longer file extensions were allowed on PCs.)

<6> 3 Points – I want to fetch a page called **`Home.html`** from the **`Tax`** directory folder of a site with **`www.MyOnlineTaxes.com`** as its host address. What is the full, complete URL of this site?

**`http://www.MyOnlineTaxes.com/Tax/Home.html`**

<7> 3 Points – Short Answer – I am drawing a picture in Windows Paint (or in PaintBrush on a Mac) that I want to use in my Web page. The image contains large regions of the same color, and contains 39 total colors, one of which I want to be treated as transparent when the image is rendered in a Web browser. Which file format (**`.BMP`**, **`.GIF`**, **`.JPG`**, or **`.PNG`**) should I save the image in? There may be more than one answer. Explain your choice(s).

Both **`.GIF`** and **`.PNG`** are good choices: they support lossless compression, at least 39 colors, and allow one of the colors to be considered transparent. (The `.BMP` format is not good because it is uncompressed. Neither is `.JPG` with its lossy compression, which "fuzzes" out the edges on cartoon-like drawings.)

<8> 15 Points – Fill in the blanks below with the correct information so that the background color of the Web page has the RGB color (0,255,255), the title is **Welcome!**, the link is to **www.cnn.com**, and the image is the file **My Pic.jpg** (resized to a width of 400 pixels, with the mouse-over text set to **Me!** and with the text **Hairy Beast** to appear if the image cannot be shown). There are four syntax errors in the code as written. Fix them.

```
<HTML>
     <HEAD>
          <TITLE>My Spiffy Web Page<\/TITLE>
     </HEAD>

     <BODY BGCOLOR="#00FFFF">
          <CENTER>
               <H1>Welcome!</H1>
               <A ~~HERF~~HREF="http://www.cnn.com">
                    <IMG ~~SCR~~SRC ="My%20Pic.jpg"
                         ALT  ="Hairy Beast"
                         WIDTH="400"
                         TITLE="Me!">
               </A>
          </CENTER>
     </BODY>
</HTML>
```

<9> 3 Points – Short Answer – In the client-server model used by the Web, fetching a page that uses a lot of graphic images requires that the browser initiate a separate connection for fetching each image file. Why is this approach better than opening a single connection to the server, fetching all the related files over that connection, and only closing the connection when all transfers are complete?

The client-server model does not hog the connection. It allows other requests to be handled between each of the image requests for the page.

<10> 5 Points – (1 point each):

1. If it is **7:00am** here in Massachusetts, should I download a large file from a server in Wyoming or from a server in England?

   **Wyoming** (it is 5:00am there).

2. If it is **7:00pm** here in Massachusetts, should I download a large file from a server in Wyoming or from a server in England?

   **England** (it is several hours ahead there, close to midnight, and traffic will be low).

3. Even though they are in the same time zone, should I download a large file from a server in Vermont or from a server in Florida?

   **Vermont** (the server is physically closer than the one in Florida, and is likely to have fewer intermediate "hops" and less competing traffic).

4. Yes or No: can I run a Web browser (Firefox, IE, Safari), a telnet session (PuTTY, ssh), and an FTP session (WinSCP, Fugu) over the same Internet connection at the same time?

   **Yes** (packets to and from all these programs are interleaved on the same connection).

5. Yes or No: can I run multiple telnet sessions (PuTTY, ssh) at the same time, each connected to a different remote system?

   **Yes** (I often have simultaneous connections open to two or more different servers, where I must log in with a unique username and password to each one).

<11> 3 Points – Short Answer – Why do I use PuTTY on the PC, or ssh in Terminal on a Mac, instead of a "traditional" telnet program?

**Because their communications are encrypted.** Even if my packets are intercepted, a "bad guy" can't extract my username or password from them. This is not true with traditional telnet programs, which send such sensitive information in the clear.

## *BÉZIER CURVES*

<12> 3 Points – Short Answer – The curve tool in Windows Paint is really a Bézier curve. If I join two Bézier curves end-to-end, how do I make one curve blend smoothly into the second?

**Make the common endpoints and their corresponding control points all collinear** (in a straight line), in the order control point-common endpoints-control point. This makes the two curves share the tangent line (each curve ends up at its endpoint tangent to the line between that endpoint and its corresponding control point).

## *WORD PROCESSING*


<13> 5 Points – Examine the picture of the sign to the right for the following questions.

1. Does the typeface in the sign have serifs, or is it sans-serif?

   **Sans-serif.**

2. The letters are each around 2½ inches high. How many points is that?

   2½ inches × 72 points per inch = **180 points**.

3. On the picture, indicate by circling all the places where kerning has been ***strongly*** applied (there are a couple of places where it is questionable whether or not kerning has been applied – ignore those).

   **The obvious places are the AT pair in LOCATIONS, and the AT and LY pairs in IMMEDIATELY.** Other places where kerning *may* have been applied are in the LO pair in LOCATIONS and in the TO pair in CUSTOMS and TO, but if kerning has been applied there it is so weak as to be nearly non-existent.

<14> 3 Points – Short Answer – I put a **`.PDF`** file of a word processing document on the Web. Why would I do that, instead of just putting the **`.DOCX`** file created by Word on the Web?

1. **Adobe Acrobat Reader is free, so it is guaranteed that everyone should be able to read the document.**

2. **It is not necessary for the person reading the document to have a copy of the program from which it was created** (i.e., you don't have to have Word, just Acrobat Reader)**.**

3. **The document creator can set the security on the document to prevent modification or content extraction** (thus protecting their intellectual property)**.**

<15> 3 Points – Short Answer – Few people use style sheets for casual documents, and simple word processors such as WordPad do not support style sheets at all. Even if they are available, why should I use them? It takes considerable effort to learn to use and manage style sheets, so what is so compelling about them that makes a person want to use them?

**Style sheets make it easy to change a paragraph's format by changing the definition of its style** (no manual reformatting of the paragraph)**.**

**You can create custom styles to match the way you write.**

## *SPREADSHEETS*

<16> 5 Points – Cell **T100** contains the following formula which is then copied to cell **Z103**. What is the resulting formula in cell **Z103** after the copy has been completed?

**=R90*95-V110+$Z$99-AB98+W$105**

**=___*95-____+$Z$99-____+_$105** (partial credit)

**=X93*95-AB113+$Z$99-AH101+AC$105** (full credit)

Relative to **T100**, cell **Z103** is offset by (right 6, down 3). Applying that to each relative cell reference gives us:

**R90** (right 6, down 3) → **X93**
**V110** (right 6, down 3) → **AB113**
**AB98** (right 6, down 3) → **AH101**
**W** (right 6, ~~down 3~~) → **AC** (row is absolute and doesn't change)

<17> 5 Points – Here is a spreadsheet containing numeric data representing a quiz taken by three students. The professor wants to scale all the grades upwards by 5% (represented by the number in cell **B2**). Write a formula for cell **E5** to scale Fred's quiz score appropriately, but write the formula so that it can be copied down to cells **E6** and **E7** and have it change correctly to compute the scale increase for Sam and Mary.

| | A | B | C | D | E | F |
|---|---|---|---|---|---|---|
| 1 | | | | | | |
| 2 | | 1.05 | **Scale** | | | |
| 3 | | | | | | |
| 4 | | | | **Quiz 1** | **Scaled** | |
| 5 | | | **Fred** | 79 | | |
| 6 | | | **Sam** | 37 | | |
| 7 | | | **Mary** | 92 | | |

**=D5*$B$2**

The **D5** needs to be relative so that it will change to **D6** and **D7** when the formula is copied down to **E6** and **E7**, respectively, but **$B$2** must be absolute so that it does not change during the copy. The formula **=D5*1.05** will generate the correct result, but is illegal in this problem because it does not use the **B2** cell in any fashion.

<18> 3 Points – Short Answer – In the previous problem, why would it be a **bad idea** to embed the 1.05 number into the formula you wrote for cell **E5**?

**In order to change the amount of the scale, you would have to edit the formulae in cells E5, E6, and E7. By having the scale in cell B2 instead, that one value is all that requires change.** As discussed earlier, the formula **=D5*1.05** will generate the correct result, but is illegal in this problem because it does not use the **B2** cell in any fashion.

<19> 4 Points – Completely parenthesize the following expression to show exactly how it will be evaluated in Excel:

```
=  ((6  +  ((3  *  (2  ^  5))  /  9))  -  (4  *  8))
```

The operator with the highest level precedence is the exponentiation **^** operator, so it gets parenthesized first. Next are multiplication ***** and division **/** operators, and because there are several such in a row they get performed from left-to-right. The parentheses for these operators are shown in red.

After **^**, *****, and **/** are performed, the remaining operators are addition **+** and subtraction **-**, and because there are several of those in a row they, too, get performed from left-to-right. The parentheses for these operators are shown in blue.