CMSI 284 Encoding Exercise
or, the Joy of Hex

Submitted by: Harris Lummis

Instructions

Do all of these problems *without* the aid of a computer. The purpose of these exercises is for you to develop “manual” encoding skills, which you will need in the event that a zombie apocalypse wipes out all known systems technology on the planet.

You may submit this assignment in one of these ways. With *both* options, make sure to show your work where work needed to be done. This provides evidence that you did not use a computer to determine the answers.

- If you know LaTeX sufficiently, copy the *source file* of this exercise and add your solutions to this copy. Commit and push the file to your GitHub repository. *Advantage:* Drop-dead clear, sharp, unambiguous presentation. *Disadvantage:* Intermediate computations may be harder to write down.

- Alternatively, you may *print* the PDF version of this exercise and do your work on paper. Submit this printout with your *name* in the designated blank up top. *Advantage:* More convenient for showing your work. *Disadvantage:* Handwritten answers may be harder to read.

Mapping to Outcomes and Proficiencies

The overall assignment covers outcomes $1a$, $1b$, $4d$, and $4f$. Each question will pertain specifically to either $1a$ or $1b$ and will be given a score ranging from 0 to 4 based on the correctness of the answer. The average score for a given outcome, rounded, determines the final proficiency for the assignment. e.g., If your numeric encoding answers attain an average of 3.2, then $1a$ will get a proficiency of |.

Outcome $4d$ will be determined by how well you use the information given in class to compute the requested answers, and how accurately you follow the instructions in this assignment.

Outcome $4f$ will be determined by whether you submit the assignment on time.

1 Integers

Outcome 1a: Assuming a 16-bit storage word, choose a value in the requested encoding and specification, then provide its corresponding values for the other encodings:

For the following problems, these powers of 2 may come in handy:

x	2^x
10	1024
11	2048
12	4096
13	8192
14	16384
15	32768
16	65536

1. Signed decimal < -31000, not divisible by 2, 4, or 8:

   -31567

   (a) Unsigned decimal:

   1000 0100 1011 0001

$$\begin{aligned}
1 \times 2 \times 2 \times 2 &= 8 \\
(8 \times 2 \times 2 + 1) \times 2 \times 2 &= 132 \\
132 \times 2 + 1 &= 265 \\
265 \times 2 &= 530 \\
530 \times 2 + 1 &= 1061 \\
1061 \times 2 + 1 &= 2123 \\
2123 \times 2 &= 4246 \\
4246 \times 2 &= 8492 \\
8492 \times 2 &= 16984 \\
16984 \times 2 + 1 &= \boxed{33969}
\end{aligned}$$

   (b) Hexadecimal:

   Use a hex digit to represent each four digit block of signed binary integer

1000	0100	1011	0001
8	4	B	1

   $\boxed{84B1}$

(c) Binary:

$$\begin{array}{r|l} 2\,\overline{)31567} & 1 \\ 2\,\overline{)15783} & 1 \\ 2\,\overline{)7891} & 1 \\ 2\,\overline{)3945} & 1 \\ 2\,\overline{)1972} & 0 \\ 2\,\overline{)986} & 0 \\ 2\,\overline{)493} & 1 \\ 2\,\overline{)246} & 0 \\ 2\,\overline{)123} & 1 \\ 2\,\overline{)61} & 1 \\ 2\,\overline{)30} & 0 \\ 2\,\overline{)15} & 1 \\ 2\,\overline{)7} & 1 \\ 2\,\overline{)3} & 1 \\ 2\,\overline{)1} & 1 \end{array}$$

0111 1011 0100 1111
Flip all bits after rightmost 1
$\boxed{\text{1000 0100 1011 0001}}$

2. Hexadecimal between `A234` and `DFFF` inclusive, no zeroes:

CAFE

(a) Unsigned decimal:

$$\begin{aligned} 1 \times 2 + 1 &= 3 \\ 3 \times 2 \times 2 &= 12 \\ 12 \times 2 + 1 \times 2 &= 50 \\ 50 \times 2 + 1 \times 2 &= 202 \\ 202 \times 2 + 1 &= 405 \\ 405 \times 2 + 1 &= 811 \\ 811 \times 2 + 1 &= 1623 \\ 1623 \times 2 + 1 &= 3247 \\ 3247 \times 2 + 1 &= 6495 \\ 6495 \times 2 + 1 &= 12991 \\ 12991 \times 2 + 1 &= 25983 \\ 25983 \times 2 &= \boxed{51966} \end{aligned}$$

(b) Signed decimal:
1100 1010 1111 1110
First bit is 1, so number is negative. Flip all bits after first 1.
0011 0101 0000 0010

$$\begin{aligned}
1 \times 2 + 1 &= 3 \\
3 \times 2 &= 6 \\
6 \times 2 + 1 &= 13 \\
13 \times 2 &= 26 \\
26 \times 2 + 1 &= 53 \\
53 \times 2 &= 106 \\
106 \times 2 &= 212 \\
212 \times 2 &= 424 \\
424 \times 2 &= 848 \\
848 \times 2 &= 1696 \\
1696 \times 2 &= 3392 \\
3392 \times 2 + 1 &= 6785 \\
6785 \times 2 &= 13570 \\
-1 \times 13570 &= \boxed{-13570}
\end{aligned}$$

(c) Binary:

C	A	F	E
1100	1010	1111	1110

$\boxed{\text{1100 1010 1111 1110}}$

3. Hexadecimal between `0111` and `01FF` inclusive, two zeroes max:

   01E0

   (a) Unsigned decimal:
   The number is positive in a signed representation, so its unsigned representation will be equivalent. $\boxed{480}$

(b) Signed decimal:

Using binary representation, first digit is a 0 so number is positive.

$$\begin{aligned} 1 \times 2^8 &= 256 \\ 256 + 2^7 &= 384 \\ 384 + 2^6 &= 448 \\ 448 + 2^5 &= \boxed{480} \end{aligned}$$

(c) Binary:

Encode each Hex digit using 4 bits:

0	1	E	0
0000	0001	1110	0000

$\boxed{\text{0000 0001 1110 0000}}$

4. Binary with high-order bits `1011` and at least 5 `1`s:

1011 0111

(a) Unsigned decimal:

1011 0111

$$\begin{aligned} 1 \times 2^7 &= 128 \\ 128 + 2^5 &= 160 \\ 160 + 2^4 &= 176 \\ 176 + 2^2 &= 180 \\ 180 + 2^1 &= 182 \\ 182 + 2^0 &= \boxed{183} \end{aligned}$$

(b) Signed decimal:

First bit is a 1, so number is negative. Flip all bits to the left of the rightmost 1:

1011 0111
0100 1001

$$\begin{aligned} 1 \times 2^6 &= 64 \\ 64 + 2^3 &= 72 \\ 72 + 2^0 &= \boxed{73} \end{aligned}$$

(c) Hexadecimal:

One hex digit represents each 4 binary digits: $\begin{array}{cc} 1011 & 0111 \\ \text{B} & 7 \end{array}$

$\boxed{B7}$

5. Hexadecimal between `8000` and `A000` *exclusive*, one zero max:

   890B

   (a) Unsigned decimal:

$$\begin{aligned} 8 \times 16^3 &= 8 \times 4096 \\ 8 \times 4096 &= 32768 \\ 32768 + 9 \times 256 &= 32768 + 2304 \\ 32768 + 2304 &= 35072 \\ 35072 + 11 &= \boxed{35083} \end{aligned}$$

   (b) Signed decimal:
   First digit is a 1, so number is negative. Flip all bits left of rightmost 1.

1000	1001	0000	1011
0111	0110	1111	0101

$$\begin{aligned} (1 \times 2 + 1) \times 2 + 1 &= 7 \\ 7 \times 2 \times 2 + 1 &= 29 \\ (29 \times 2 + 1) \times 2 &= 118 \\ (118 \times 2 + 1) \times 2 + 1 &= 475 \\ 475 \times 2 + 1 &= 951 \\ 951 \times 2 + 1 &= 1903 \\ 1903 \times 2 &= 3806 \\ 3806 \times 2 + 1 &= 7613 \\ 7613 \times 2 &= 15226 \\ 15226 \times 2 + 1 &= 30453 \end{aligned}$$

$$\boxed{-30453}$$

   (c) Binary:
   Use 4 bits to represent each hex digit

8	9	0	B
1000	1001	0000	1011

$\boxed{1000\ 1001\ 0000\ 1011}$

6. Unsigned decimal between 48000 and 65000 inclusive, not divisible by 4 or 8:

   48391

   (a) Signed decimal:

   1011 1101 0000 0111

   First bit is a 1, number is negative. Flip all bits left of rightmost 1.

   1011 1101 0000 0111

   0100 0010 1111 1001

$$\begin{aligned}1 \times 2^{14} &= 16384\\ 16384 + 2^9 &= 16896\\ 16896 + 2^7 &= 17024\\ 17024 + 2^6 &= 17088\\ 17088 + 2^5 &= 17120\\ 17120 + 2^4 &= 17136\\ 17136 + 2^3 &= 17144\\ 17144 + 2^0 &= 17145\\ &\boxed{-17145}\end{aligned}$$

   (b) Hexadecimal:

   Use a hex digit to represent each nybble:

1011	1101	0000	0111
B	D	0	7

   $\boxed{BD07}$

   (c) Binary:

$$
\begin{array}{rl}
2\,\overline{)48391} & 1 \\
2\,\overline{)24195} & 1 \\
2\,\overline{)12097} & 1 \\
2\,\overline{)6048} & 0 \\
2\,\overline{)3024} & 0 \\
2\,\overline{)1512} & 0 \\
2\,\overline{)756} & 0 \\
2\,\overline{)378} & 0 \\
2\,\overline{)189} & 1 \\
2\,\overline{)94} & 0 \\
2\,\overline{)47} & 1 \\
2\,\overline{)23} & 1 \\
2\,\overline{)11} & 1 \\
2\,\overline{)5} & 1 \\
2\,\overline{)2} & 0 \\
2\,\overline{)1} & 1
\end{array}
$$

$\boxed{1011\ 1101\ 0000\ 0111}$

7. Unsigned decimal between 80 and 1024 *exclusive*, not divisible by 4 or 8:

   93

   (a) Signed decimal:

   First bit is a zero, so number is positive, thus unsigned and signed representations are identical in decimal value. $\boxed{93}$

   (b) Hexadecimal:

   Use a hex digit to represent each nybble:

0000	0000	0101	1101
0	0	5	D

   $\boxed{005D}$

(c) Binary:

$$\begin{array}{r|l}
2\,\overline{)93} & 1 \\
2\,\overline{)46} & 0 \\
2\,\overline{)23} & 1 \\
2\,\overline{)11} & 1 \\
2\,\overline{)5} & 1 \\
2\,\overline{)2} & 0 \\
2\,\overline{)1} & 1
\end{array}$$

$\boxed{0000\ 0000\ 0101\ 1101}$

8. Signed decimal between -69 and -192 inclusive, not divisible by 2, 4, or 8:

   -143

   (a) Unsigned decimal:

$$\begin{aligned}
15 \times 16^3 &= 61440 \\
61440 + 15 \times 16^2 &= 65280 \\
65280 + 7 \times 16 &= 65392 \\
65392 + 1 &= \boxed{65393}
\end{aligned}$$

   (b) Hexadecimal:

   Use a hex digit to represent each nybble:

1111	1111	0111	0001
F	F	7	1

   $\boxed{\text{FF 71}}$

   (c) Binary:

$$\begin{array}{r|l}
2\,\overline{)143} & 1 \\
2\,\overline{)71} & 1 \\
2\,\overline{)35} & 1 \\
2\,\overline{)17} & 1 \\
2\,\overline{)8} & 0 \\
2\,\overline{)4} & 0 \\
2\,\overline{)2} & 0 \\
2\,\overline{)1} & 1
\end{array}$$

   Take two's complement:

0000	0000	1000	1111
1111	1111	0111	0001

$\boxed{1111\ 1111\ 0111\ 0001}$

9. Binary with high-order bits `0001` and at least `7 1`s:

   0001 1101 1100 1100

   (a) Unsigned decimal: Signed is positive, unsigned will be same value: $\boxed{7628}$

   (b) Signed decimal:
   First bit is a 0, will be same as unsigned rep:

$$\begin{aligned} 16^3 &= 4096 \\ 4096 + 13 \times 16^2 &= 7424 \\ 7427 + 12 \times 16 &= 7616 \\ 7616 + 12 &= \boxed{7628} \end{aligned}$$

   (c) Hexadecimal:

0001	1101	1100	1100
1	D	C	C

$\boxed{\text{1D CC}}$

10. Hexadecimal between `284C` and `789A` *exclusive*, one zero max:

    365B

    (a) Unsigned decimal:

    First bit is a 0, will be same as signed decimal: $\boxed{13915}$

    (b) Signed decimal:

$$\begin{aligned} 3 \times 16^3 &= 12288 \\ 12288 + 6 \times 16^2 &= 13824 \\ 13824 + 5 \times 16 &= 13904 \\ 13904 + 11 &= \boxed{13915} \end{aligned}$$

(c) Binary:

3	6	5	B
0011	0110	0101	1011

$\boxed{0011\ 0110\ 0101\ 1011}$

2 Negation

Outcome 1a: Choose 16-bit signed words according to the given specifications, then compute their negatives, expressing your answers in hex as well. You may have a maximum of eight hex `0` digits among your chosen values:

1. x in [`9876…CDEF`] = ABCD; $-x$ = 5433

A	B	C	D
1010	1011	1100	1101

   Flip bits left of rightmost 1:

1010	1011	1100	1101
0101	0100	0011	0011

   Use a hex digit to represent each nybble:

0101	0100	0011	0011
5	4	3	3

2. y in [`D219…EDEE`] = E3F0; $-y$ = 1C10

E	3	F	0
1110	0011	1111	0000

   Flip all bits left of rightmost 1:

1110	0011	1111	0000
0001	1100	0001	0000

   Use a hex digit to represent each nybble:

0001	1100	0001	0000
1	C	1	0

3. z is odd, in [`8087…9191`] = 8F03; $-z$ = 70FD

8	F	0	3
1000	1111	0000	0011

1000	1111	0000	0011
0111	0000	1111	1101

0111	0000	1111	1101
7	0	F	D

4. w is even, in [`3BB0...5FFE`] = 4BDC; $-w$ = B424

4	B	D	C
0100	1011	1101	1100

0100	1011	1101	1100
1011	0100	0010	0100

1011	0100	0010	0100
B	4	2	4

5. m in [`010A...020B`] = 01BF; $-m$ = FE41

0	1	B	F
0000	0001	1011	1111

0000	0001	1011	1111
1111	1110	0100	0001

1111	1110	0100	0001
F	E	4	1

3 Signed Arithmetic

Outcome 1a: Choose 16-bit addends with the given specifications and compute the requested sums and states using **signed** arithmetic. You may have a maximum of twelve hex `0` digits among your chosen values (not including the ones already given):

1. `2 8 C 6` + `5 A 3 1`

   (a) Sum, saturated:

2	8	C	6
+5	A	3	1
8	2	F	7

   Addition of two positives yields a negative result, which indicates overflow.
   $\boxed{\text{7FFF}}$

   (b) Sum, modular:

   $\boxed{\text{82F7}}$

(c) Carry (y/n): N

(d) Overflow (y/n): Y

2. `7 0 F 2` + `E 7 C D`

   (a) Sum, saturated:

	7	0	F	2
+	E	7	C	D
1	5	8	B	F

   $\boxed{\text{58BF}}$

   (b) Sum, modular:
   $\boxed{\text{58BF}}$

   (c) Carry (y/n): Y

   (d) Overflow (y/n): N

3. `B B A A` + `8 A C E`

   (a) Sum, saturated:

	B	B	A	A
+	8	A	C	E
1	4	7	7	8

   $\boxed{8000}$

   (b) Sum, modular:
   $\boxed{4778}$

   (c) Carry (y/n):Y

   (d) Overflow (y/n):Y

4. `A 1 4 B` + `9 A 7 0`

   (a) Sum, saturated:

	A	1	4	B
+	9	A	7	0
1	3	B	B	B

   Two negatives result in a positive, overflow
   $\boxed{8000}$

(b) Sum, modular:
$\boxed{3BBB}$

(c) Carry (y/n):Y

(d) Overflow (y/n):Y

5. `A 2 1 9` + `2 A D E`

(a) Sum, saturated:

$$\begin{array}{ccccc} & A & 2 & 1 & 9 \\ + & 2 & A & D & E \\ \hline & C & C & F & 7 \end{array}$$

$\boxed{CCF7}$

(b) Sum, modular: $\boxed{CCF7}$

(c) Carry (y/n):N

(d) Overflow (y/n):N

6. `5 8 0 0` + `0 F A A`

(a) Sum, saturated:

$$\begin{array}{ccccc} & 5 & 8 & 0 & 0 \\ + & 0 & F & A & A \\ \hline & 6 & 7 & A & A \end{array}$$

$\boxed{67AA}$

(b) Sum, modular:
$\boxed{67AA}$

(c) Carry (y/n):N

(d) Overflow (y/n):N

7. `C 0 0 0` + `C 0 0 0`

(a) Sum, saturated:

$$\begin{array}{ccccc} & C & 0 & 0 & 0 \\ + & C & 0 & 0 & 0 \\ \hline 1 & 8 & 0 & 0 & 0 \end{array}$$

Two negatives, negative result, no overflow
$\boxed{8000}$

(b) Sum, modular:
$\boxed{8000}$

(c) Carry (y/n):Y

(d) Overflow (y/n):N

4 Units of Information

Outcome 1a, 6 answers: Many storage manufacturers sell the same product at different capacities (e.g., Western Digital My Book; Drobo storage array; SanDisk SDXC Memory Card). Go window shopping and find product listings for the smallest- and largest-capacity versions of such a product.

1. (not graded; mainly for reference) Provide the brand, model, min/max capacities, and prices of the product line you've chosen:

   My Book Duo:
   12TB - $499.99 (promo price)
   4TB - $249.99 (promo price)

2. Interpret the device capacities as decimal units (i.e., megabytes, gigabytes, terabytes, etc.). Show your calculations to answer the following:

   (a) How much does a kilobyte cost on the smallest-capacity version of the device?

   $$\frac{\$249.99}{4 \times 10^9 \text{ kb}} = \boxed{\frac{\$6.24975 \times 10^{-8}}{\text{kb}}}$$

   (b) How much does a kilobyte cost on the largest-capacity version of the device?

   $$\frac{\$499.99}{12 \times 10^9 \text{ kb}} = \boxed{\frac{\$4.16658 \times 10^{-8}}{\text{kb}}}$$

   (c) What is the price difference, on a per-kilobyte basis, between the smallest- and largest-capacity versions of the device?

   $$\$6.24975 \times 10^{-8} - \$4.16658 \times 10^{-8} = \boxed{\$2.083167 \times 10^{-8}}$$

3. Interpret the device capacities as binary units (i.e., mebibytes, gibibytes, tebibytes, etc.). Show your calculations to answer the following:

(a) How much does a kibibyte cost on the smallest-capacity version of the device?

$$\frac{\$249.99}{4 \times 2^{30} \text{ kb}} = \boxed{\frac{\$5.82053 \times 10^{-8}}{\text{kb}}}$$

(b) How much does a kibibyte cost on the largest-capacity version of the device?

$$\frac{\$499.99}{12 \times 2^{30} \text{ kb}} = \boxed{\frac{\$3.88043 \times 10^{-8}}{\text{kb}}}$$

(c) What is the price difference, on a per-kibibyte basis, between the smallest- and largest-capacity versions of the device?

$$\$5.82053 \times 10^{-8} - \$3.88043 \times 10^{-8} = \boxed{\$1.9401 \times 10^{-8}}$$

5 IEEE 754 Encoding

Outcome 1a, 8 answers: Read each question carefully and provide the requested answers using the proper encoding:

1. Choose a number between 0 and 1 that has at least 4 non-zero digits in the decimal and is *not* a power of 2 (e.g., 0.0625 is 2^{-4} and thus would not count):

   Your chosen number in decimal form: 0.5625

   (a) Single-precision (32-bit) approximation:
   Sign is positive, first bit is 0.

$$\begin{aligned} 0.5625 \times 2 &= 1.125 && 1 \\ 0.125 \times 2 &= 0.25 && 0 \\ 0.25 \times 2 &= 0.5 && 0 \\ 0.5 \times 2 &= 1.0 && 1 \\ & && = 0.1001 \end{aligned}$$

Normalize:

$$\begin{aligned} 0.1001 &= 1.001 \times 2^{-1} \\ f &= 001\ 0000\ 0000\ 0000\ 0000\ 0000 \end{aligned}$$

Add bias to exponent of two, place in exponent field:

$$\begin{aligned} 2^{8-1} - 1 + (-1) &= 126 \\ (126)_{10} &= 0111\ 1110 \end{aligned}$$

Put it all together

$\boxed{0\ 0111\ 1110\ 001\ 0000\ 0000\ 0000\ 0000\ 0000}$

(b) Double-precision (64-bit) approximation:
Steps identical until decision of f:

$$f = 0010\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000$$

$$\begin{aligned} 2^{11-1} - 1 + (-1) &= 1022 \\ (1022)_{10} &= 01111111110 \end{aligned}$$

0011 1111 1110 0010 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000

$\boxed{\text{3FE2 0000 0000 0000}}$

2. Determine the smallest positive whole number that cannot be represented in memory with the given floating point encoding, and state why:

   (a) ...in single-precision (32-bit):
   $s = 0,\ \text{e} = 1111\ 1110,\ \text{f} = 111\ 1111\ 1111\ 1111\ 1111$

$$\begin{aligned} (1 + (f \times 2^{-23})) \times 2^{23} &= 111111111111111111111111 \\ 111111111111111111111111 + 10 &= \boxed{1 \times 2^{24} + 1} \end{aligned}$$

This number cannot be represented in 32-bit IEEE as the least significant bit is too many orders of magnitude away from the most significant bit to be represented in f.

(b) ...in double-precision (64-bit):

$s = 0,$ e $=$ 111 1111 1110,

f $=$ 1111 1111 1111 1111 1111 1111 1111 1111 1111 1111 1111 1111 1111

Using the same model as above, :

$$(1 + (f \times 2^{-52})) \times 2^{52} = 1111111111111111111111111111111111111111111111111111$$

$$111111111111111111111111 + 10 = \boxed{1 \times 2^{53} + 1}$$

This number cannot be represented in 64-bit IEEE as the least significant bit is too many orders of magnitude away from the most significant bit to be represented in f.

3. Choose an 8-digit hexadecimal number where *no* digit is repeated nor sequential (e.g., `134F EA85` is not allowed because of `F` followed by `E`):

`472A 6B31`

(a) Provide the IEEE 754 floating-point value that these bits represent in base 2 (use scientific notation):

0100 0111 0010 1010 0110 1011 0011 0001

$s = 0,\ e = 1000\ 1110,\ f = 010\ 1010\ 0110\ 1011\ 0011\ 0001$

$$\begin{aligned} e &= 142 \\ f \times 2^{-23} + 1 &= 1.010\ 1010\ 0110\ 1011\ 0011\ 0001 \\ ((f \times 2^{-23}) + 1) \times 2^{(142-127)} &= 1010\ 1010\ 0110\ 1011\ .\ 0011\ 0001 \\ &= \boxed{1.01010100110101100110001 \times 2^{15}} \end{aligned}$$

(b) Provide its closest approximate value in base 10 (use scientific notation if necessary):

Using Dorin's method, which employs elementary school techniques that I now know need not be shown, get decimal number left of radix point = 43627

Add to this:

$$43627 + \frac{1}{2^3} + \frac{1}{2^4} + \frac{1}{2^8} = 43627.1252441797$$
$$= \boxed{4.36271252441797 \times 10^4}$$

4. Choose 16-digit hexadecimal number where no more than 2 adjacent digits are the same (e.g., `0102 EFF3 E157 C411` is not allowed because of `FF` and `11`):

`0352 1126 89A2 BC03`

(a) Provide the IEEE 754 floating-point value that these bits represent in base 2 (use scientific notation):

Convert to binary:

`0000 0011 0101 0010 0001 0001 0010 0110 1000 1001 1010 0010 1011 1100 0000 0011`

$s = 0, e =$ `000 0011 0101`,
$f =$ `0010 0001 0001 0010 0110 1000 1001 1010 0010 1011 1100 0000 0011`

$e < 2046, s = 0$ so number is positive:

$e = 16 + 32 + 5 = 53$
$e - 1023 = -970$

$(1+f\times2^{-52})\times2^{(-970)} = \boxed{1.0010000100010010011010001001101000101011110000000011 \times 2^{-970}}$

(b) Provide its closest approximate value in base 10 (use scientific notation if necessary):

$\boxed{\frac{1}{2^{970}} + \frac{1}{2^{973}} + \frac{1}{2^{978}} + ...}$

6 Character Encoding

Outcome 1b, 44 answers: Read each question carefully and provide the requested answers using the proper encoding. *Remember to show your work to prove that you encoded these* *manually:*

1. The city of Los Angeles is unhappy with the Unicode SNOWMAN character, as they are unable to use it on their official documents to represent a fun day in the snow. They requested a new "sandman" symbol in its place.

   Pick a codepoint for this new character, and show how it would be encoded in UTF-8, UTF-16, and UTF-32. The codepoint must be have 6 unique digits and start with a `10` (i.e., it belongs under Supplemental Private Use Area-B).

   `103BDE`

   Binary rep: 0001 0000 0011 1011 1101 1110

   (a) UTF-8:
   Strip leading zeroes and respace digits:
   100 000011 101111 011110

   Add encoding bits:
   11110100 10000011 10101111 10011110

   Convert to Hex:
   $\boxed{\text{F4 83 AF 9E}}$

   (b) UTF-16:

   $$\begin{array}{cccc} & 10 & 3B & DE \\ - & 01 & 00 & 00 \\ \hline & 0E & 3B & DE \end{array}$$

   Convert to binary:
   0000 1110 0011 1011 1101 1110

   Remove leading zeros and respace bits:
   11 10001110 11 11011110

   Add in encoding bits:
   11011011 10001110 11011111 11011110

   Convert to hex:
   $\boxed{\text{DB 8E DF DE}}$

   (c) UTF-32:
   $\boxed{\text{00 10 3B DE}}$

2. Encode the first eight letters of your first and last name combined (including the space in between) as requested, *replacing four of them* with corresponding characters from the Enclosed Alphanumerics Unicode block (uppercase or lowercase, your choice):

   First 8 characters including space: Harris L - Replacing H, both r, and L (

   (a) UTF-8:

       U+24BD U+0061 U+24E1 U+24E1 U+0069 U+0073 U+0020 U+24C1

       (H) → 0010 0100 1011 1101 → 1110 0010 1001 0010 1011 1101 → E2 92 BD
       a → 0000 0000 0110 0001 → 0110 0001 → 61
       (r) → 0010 0100 1110 0001 → 1110 0010 1001 0011 1010 0001 → E2 93 A1
       i → 0000 0000 0110 1001 → 0110 1001 → 69
       s → 0000 0000 0111 0011 → 0111 0011 → 73
       space → 0000 0000 0010 0000 → 0010 0000 → 20
       (L) → 0010 0100 1100 0001 → 1110 0010 1001 0011 1000 0001 → E2 93 81

       $\boxed{\text{E2 92 BD 61 E2 93 A1 E2 93 A1 69 73 20 E2 93 81}}$

   (b) UTF-16:

       No characters have codepoint > FFFF, so all encodings are still 16 bits

       U+24BD U+0061 U+24E1 U+24E1 U+0069 U+0073 U+0020 U+24C1

       $\boxed{\text{24 BD 00 61 24 E1 24 E1 00 69 00 73 00 20 24 C1}}$

   (c) UTF-32:

       $\boxed{\text{0000 24BD 0000 0061 0000 24E1 0000 24E1 0000 0069 0000 0073 0000 0020 0000 24C1}}$

   (since we're dealing with Unicode anyway, if your name, when properly written, has an accent or other diacritical, then use that too)

3. Choose four emoji without variants (the monster master list can be found in `http://unicode.org/emoji/charts/full-emoji-list.html`) to describe your dream vacation. Encode them: U+26F0 U+1F3D5 U+1F304 U+1F30C

   (a) UTF-8:

Convert to Binary:

U+26F0	0010 0110 1111 0000
U+1F3D5	0001 1111 0011 1101 0101
U+1F304	0001 1111 0011 0000 0100
U+1F30C	0001 1111 0011 0000 1100

Respace bits and pad:

U+26F0	0010 011011 110000
U+1F3D5	000 011111 001111 010101
U+1F304	000 011111 001100 000100
U+1F30C	000 011111 001100 001100

Add encoding bits:

U+26F0	11100010 10011011 10110000
U+1F3D5	11110000 10011111 10001111 10010101
U+1F304	11110000 10011111 10001100 10000100
U+1F30C	11110000 10011111 10001100 10001100

Convert back to hex:

U+26F0	E2 9B B0
U+1F3D5	F0 9F 8F 95
U+1F304	F0 9F 8C 84
U+1F30C	F0 9F 8C 8C

`E2 9B B0 F0 9F 8F 95 F0 9F 8C 84 F0 9F 8C 8C`

(b) UTF-16:

Prepare for Conversion to binary:

U+26F0	26 F0
U+1F3D5	0F3D5
U+1F304	0F304
U+1F30C	0F30C

Convert all but first to binary:

U+26F0	26 F0
U+1F3D5	0000 1111 0011 1101 0101
U+1F304	0000 1111 0011 0000 0100
U+1F30C	0000 1111 0011 0000 1100

Respace bits:

U+26F0	26 F0
U+1F3D5	00 00111100 11 11010101
U+1F304	00 00111100 11 00000100
U+1F30C	00 00111100 11 00001100

Add encoding bits:

U+26F0	26 F0
U+1F3D5	1101 1000 0011 1100 1101 1111 11010101
U+1F304	1101 1000 0011 1100 1101 1111 0000 0100
U+1F30C	1101 1000 0011 1100 1101 1111 0000 1100

Convert back to hex:

U+26F0	26 F0
U+1F3D5	D8 3C DF D5
U+1F304	D8 3C DF 04
U+1F30C	D8 3C DF 0A

$\boxed{\texttt{26 F0 D8 3C DF D5 D8 3C DF 04 D8 3C DF 0A}}$

(c) UTF-32:
$\boxed{\texttt{00 00 26 F0 00 01 F3 D5 00 01 F3 04 00 01 F3 0C}}$

Fun tip: Remember that there are *flag* emoji to represent specific locations.

4. This one is given 5 times the weight: explain why `https://xkcd.com/380/` is funny. (yes, it's funny) Remember, XKCD comics include a mouseover caption that is an integral part of the strip.

   This xkcd is funny because the basilisk is still fatal on sight *even in emoji form.* Even funnier, though, is the caption, which says that the eye of the basilisk character has the codepoint U+FDD0 which is actually the unicode ”no character” symbol. Ha!