Problem 1. (4 points):

Suppose you are working on an 16 bit machine (vs 32 or 64 as we are accustomed to), and you are presented with the following bytes:

```
10010100 00110101
```

Please answer the following questions...

1. What is the decimal value of these bits if they are interpreted as an unsigned integer using little endian format?

   00110101 10010100

   32,768 16,384 8,192 4,096 2,048 1,024 512 256 128 64 32 16 8 4 2 1

   $8{,}192 + 4{,}096 + 1{,}024 + 256 + 128 + 16 + 4 = \boxed{13{,}716}$

2. What is the decimal value of these bits if they are interpreted as two's compliment signed integer using little endian format?

   00110101 10010100

   SAME AS ABOVE

   $\boxed{13{,}716}$

3. What is the decimal value of these bits if they are interpreted as an unsigned integer using big endian format?

   10010100 00110101

   4,096 1,024 32 16 4 1

   $5{,}173 + 32{,}768 = \boxed{37{,}941}$

4. What is the decimal value of these bits if they are interpreted as two's compliment signed integer using big endian format?

   −1 10010100 00110101

   32 16 4 1

   $(-32{,}768) + 4{,}096 + 1{,}024 + 32 + 16 + 4 + 1$

   $= \boxed{-27{,}595}$

Problem 2. (4 points):

Consider the following bytes represented in hexadecimal...

```
63 73 20 69 73 20 63 6f 6f 6c 00
```

Please answer the following questions... (show your work)

1. Determine and show the binary representation of these bytes. [2 points]

63	73	20	69	73	20	63	6f	6f	6c	00
0110 0011	0111 0011	0010 0000	0110 1001	0111 0011	0010 0000	0110 0011	0110 1111	0110 1111	0110 1100	0000 0000

0110 0011 0111 0011 0010 0000 0110 1001 0111 0011 0010 0000 0110 0011 0110 1111 0110 1111 0110 1100 0000 0000

2. If these bytes were interpreted as ascii characters, what string would they contain? [2 points]

cs is cool

mantissa

[illegible] $= 0.551648736$

Problem 3. (4 points):

Consider the following 32 bits...

10010100110001101001110001101101

Please answer the following questions (show your work or you will not receive credit)...

1. What is the single precision (32 bit) floating point value represented by these bits? [3 points]

negative

exponent

$32+8+1=41$

$\times 2^{(41-127)} = \times 2^{(-86)}$

$$\boxed{(-1)(1+0.551648736)(2^{-86}) = 2.00545899 \times 10^{-26}}$$

2. What is the hexadecimal representation of these bits? [1 point]

$\boxed{\text{Hex: } 0x\ 94\ c6\ 9c\ 6d}$

Problem 4. (13 points):

Consider an **9-bit** machine that supports both signed and unsigned arithmetic. A *short* integer is encoded using 5 bits; *unsigned* denotes an unsigned computation. You have the following variables:

```
short sy = -13;
int y = sy;
int x = -42;
unsigned ux = x;
```

5 9 9

1 0 0 1 1

16 8 4 2 1

Fill in all of the missing entries in the table.

Number	Decimal Representation	Binary Representation
Umin	0	0 0000 0000
Umax	511	1 1111 1111
y	−13	1 1111 0011
ux	470	1 1101 0110
Twos-Comp	−23	1 1110 1001
Twos-Comp	108	0 0110 1100
Twos-Comp	-27	1 1110 0101
x + y	-55	1 1100 1001
TMax	255	0 1111 1111
TMin	-256	1 0000 0000
TMin+TMin	0	0 0000 0000
TMin+1	-255	1 0000 0001
TMax+1	-256	1 0000 0000
−TMax	-1	1 1111 1111
−TMin	0	0 0000 0000

Problem 5. (10 points):

Consider the following program and that a `long` is 8-bytes, an `int` is 4-bytes, and a `char` is 1-byte.

```
#include <stdio.h>

#define SIZE 24

int main() {
  char str[SIZE];
  long *u_ptr = (long*)str;
  int *i_ptr = (int *)(u_ptr + 1);
  char *c_ptr = (char *)(i_ptr + 2);

  scanf("%lx %x %x %s", u_ptr, i_ptr, i_ptr + 1, c_ptr);

  printf("str = %s\n", str);
  return 0;
}
```

Write down the needed input to be sent to *scanf* so that the call to *printf* outputs

```
str = I <3 cs 224!! (*^_^*)
```

Drawing memory to keep track of what is happening is highly recommended. If you need to "code this up" and play around with it to better understand it, this is allowed. **The input is:**

736320333c2049 34323220 28202121 *^_^*)

I	49
Space	20
<	3c
3	33
Space	20
c	63
s	73
2	32
2	32
4	34
!	21
!	21
Space	20
(	28
*	2a
^	5e
_	5f
^	5e
*	2a
)	29

ASCII hexadecimal set:

00 nul	01 soh	02 stx	03 etx	04 eot	05 enq	06 ack	07 bel
08 bs	09 ht	0a nl	0b vt	0c np	0d cr	0e so	0f si
10 dle	11 dc1	12 dc2	13 dc3	14 dc4	15 nak	16 syn	17 etb
18 can	19 em	1a sub	1b esc	1c fs	1d gs	1e rs	1f us
20 sp	21 !	22 "	23 #	24 $	25 %	26 &	27 '
28 (	29)	2a *	2b +	2c ,	2d -	2e .	2f /
30 0	31 1	32 2	33 3	34 4	35 5	36 6	37 7
38 8	39 9	3a :	3b ;	3c <	3d =	3e >	3f ?
40 @	41 A	42 B	43 C	44 D	45 E	46 F	47 G
48 H	49 I	4a J	4b K	4c L	4d M	4e N	4f O
50 P	51 Q	52 R	53 S	54 T	55 U	56 V	57 W
58 X	59 Y	5a Z	5b [	5c \	5d]	5e ^	5f _
60 `	61 a	62 b	63 c	64 d	65 e	66 f	67 g
68 h	69 i	6a j	6b k	6c l	6d m	6e n	6f o
70 p	71 q	72 r	73 s	74 t	75 u	76 v	77 w
78 x	79 y	7a z	7b {	7c |	7d }	7e ~	7f del

Problem 6. (12 points):

Consider the following 11-bit floating point representation based on the IEEE floating point format. There is a sign bit in the most significant bit. The next five bits are the exponent. The last five bits are the fraction. The rules are like those in the IEEE standard including the use of a bias to encode the exponent (normalized, denormalized, representation of 0, infinity, and NAN).

As a reminder, the floating point format to encode numbers is

$$V = (-1)^s \times M \times 2^E$$

where M is the *significand* and E is the *exponent*. Fill in all the missing entries in the table below with the following instructions for each column:

Description: Some unique property of this number, such as, "The largest denormalized value."

Binary: The 11-bit representation.

M**:** The decimal value of the mantissa with our without the implied one as appropriate (e.g., binary 1.01 would be $1\frac{1}{4}$ or $\frac{5}{4}$).

E**:** The unbiased integer value of the exponent (e.g., 2^3 would be 3).

You need not fill in entries marked "—". Remember, E is unbiased and M includes the implied one for normalized values.

Description	Binary	M	E
Minus Zero	1 00000 00000	0	−14
Not a number	1 11111 11111	—	—
—	0 01101 00101	$1\frac{5}{32}$	−2
largest denormalized number	0 00000 11111	$\frac{31}{32}$	−14
—	1 00000 10011	19/32	−14
Negative one	1 01111 00000	1	0
Smallest positive normalized	0 10000 00001	1/32	−14
The value $3\frac{3}{4}$	0 10000 11100	$1\frac{7}{8}$	1
The value −1280	1 11001 01000	1 1/4	10
The value $4\frac{1}{2} \times 2^{-12}$	0 00101 00100	1 1/8	−10