UC Berkeley Fall 2018 CS61C Quest Answers

Q1a) With **3 bits**, how do we represent **-2**? If it can't be done, select "**N/A**". (Select ONE per row)

	000	001	010	011	100	101	110	111	N/A
Unsigned	○ 0	○ 1	○ 2	○ 3	○ 4	○ 5	○ 6	○ 7	●
Sign/Magnitude	○ +0	○ 1	○ 2	○ 3	○ -0	○ -1	● -2	○ -3	○
One's Complement	○ +0	○ 1	○ 2	○ 3	○ -3	● -2	○ -1	○ -0	○
Two's Complement	○ 0	○ 1	○ 2	○ 3	○ -4	○ -3	● -2	○ -1	○
Bias; use bias of $-(2^{N-1}-1)$ from lecture	○ -3	● -2	○ -1	○ 0	○ 1	○ 2	○ 3	○ 4	○

Q1b) Convert 26_{10} to the following bases (and remove any leading zeros)

Binary	*Hex*
0b11010 (**1***16+**1***8+**0***4+**1***2+**0***1=26)	**0x1A** (**1***16+10(=**A**)*1=26)

Q1c) Add these *Two's Complement* nibbles:

```
    1001
+   1011
   10100
```

Does it overflow a nibble? (Select ONE)
● **Yes [1001 (-7) + 1011 (-5) = -12. .Since 2s complement nibbles can only represent [-8,7] it certainly can't hold -12. You don't have to even do the addition to answer this correctly]**
○ No

```
int mystery (unsigned int N) {
        unsigned int counter = 0;
        while (N > 0) {
                counter += N & 1;
                N = N >> 1;
        }
        return counter == 1;
}
```

Q2a) What does the **mystery** return? (Select ONE)
○ The number of 1s in the binary representation of **N**
○ 1 if **N** is odd, otherwise 0
● **1 if N is a power of 2, otherwise 0 [it shifts N to the right, storing in counter all the 1s it sees. If it's exactly 1, meaning the only 1 is the MSB (most significant bit), then it's a power of 2]**
○ 1 if the binary representation of **N** is all 1s, otherwise 0
○ 1 if the binary representation of **N** has any 1s, otherwise 0

Q2b) Given this setup to **mystery**:
```
unsigned int myN =
GetNFromUser();
int mysteryReturn = mystery(myN);
```
...could **myN** be changed by the call to **mystery**? (Select ONE)
○ Yes
○ It depends on the value of **myN**
● **No [because C is *call by value*]**

```
// My project partner wrote this code to duplicate some elements of orig into copy
        int orig[] = {1,2,3,4,5,6,7,8}; // ints are 4 bytes wide      ...scratch space below...
        int main() {
                int *backup, *copy, **copyH;
                backup = copy = (int *) malloc (sizeof(int) * 100);
                copyH = &copy;
                for (int i = 0; i < 2; i++) {
                        *copy = orig[i];
                        *copyH = *copyH + 4;
                }
        }
```

Q3a) Right *before* the **for** loop, where in memory do the following **point**? (Select ONE per row)

	Code	Static	Stack	Heap
orig	○	●	○	○
backup	○	○	○	●
copyH	○	○	●	○

Q3b) Right *after* the **for** loop, what is the value of the following? If it'd be garbage, write "**G**".

backup[0]	backup[1]	copy[0]	copy[1]
1	G	G	G

backup,copy   **backup = copy = (int *) malloc (sizeof(int) * 100);**
↓   *the memory below shows all the words (not bytes) of the malloced space*

G	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G

backup,copy   ***copy = orig[0];**
↓

1	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G

backup   copy   ***copyH = *copyH + 4;**
↓   ↓

1	G	G	G	G	G	G	G	G	G	G	G	G	G	G	G

backup   copy   ***copy = orig[1];**
↓   ↓

1	G	G	G	2	G	G	G	G	G	G	G	G	G	G	G

backup   copy   ***copyH = *copyH + 4;**
↓   ↓

1	G	G	G	2	G	G	G	G	G	G	G	G	G	G	G
backup[0]	backup[1]							copy[0]	copy[1]						

Q4a) Which RISC-V snippet could be the compilation of the C code: **x15 = 20 - x5**? (Select ALL that apply)
Assume the C variables x5 and x15 map directly to the registers of the same name.

☐	☐	■	■
sub x5, 20, x15 *[no, sub has no immediates]*	sub x15, 20, x5 *[no, sub has no immediates]*	addi x15, x0, 20 #x15=20 sub x15, x15, x5 #x15=20-x5	addi x15,x5,-20 #x15=x5-20 sub x15, x0, x15#x15=20-x5

Q4b) Say we have an **int array A[99]** starting at address **0x00010000**, and register **x5** contains **&A[0]**.
Assuming **sizeof(int) == 4**, what value is in register **x10** after **lw x10, 8(x5)** ? (Select ALL that apply)

■	☐	☐	☐	☐	☐
A[2] 8 bytes from A is 2 ints over, thus A[2]	A[8]	&A[2]	&A[8]	0x00010008	0x00080000