Hexadecimal Addition Calculator

This is a **warmup** exercise. It is **not compulsory**, and may be completed **individually or with your lab partner**.

Printf and Hexadecimal Numbers

The `printf` function is very complex and can be used to display data in a number of different ways.

Display Width

When displaying numbers with `printf`, you can tell it how many characters to use by putting a number before the **d** in the **%d**.

For example:

```
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[]) {
    printf("%5d\n", 20);
    printf("%10d\n", 20);
    printf("%20d\n", 20);

    return EXIT_SUCCESS;
}
```

Would print the following:

```
   20
        20
                  20
```

Leading Zeroes

You can tell `printf` to fill the empty space before a value with `0` s instead of blank space by putting a `0` before the number.

For example:

```
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[]) {
    printf("%05d\n", 20);
    printf("%010d\n", 20);
    printf("%020d\n", 20);

    return EXIT_SUCCESS;
}
```

Would print the following:

```
00020
0000000020
00000000000000000020
```

Hexadecimal

The computer and `printf` store all numbers in binary, but when you use **%d** in `printf` , you're letting `printf` know that you want it to show you the number in *decimal* (base 10). You can also tell `printf` to show you numbers in *hexadecimal* (base 16) using **%x** or **%X**. **%x** will display the letters A, B, C, D, E, and F in lower-case when they appear in the number and **%X** will use upper-case. You can also use **%#X** and **%#x** to show the hexadecimal number starting with a `0x` .

For example, the following program:

```c
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[]) {
    printf("%x\n", 170);
    printf("%X\n", 170);
    printf("%#x\n", 170);
    printf("%#X\n", 170);
    printf("\n");

    printf("%x\n", 48879);
    printf("%X\n", 48879);
    printf("%#x\n", 48879);
    printf("%#X\n", 48879);
    printf("\n");

    // Adding in width
    printf("%4x\n", 170);
    printf("%04x\n", 170);
    printf("%16x\n", 170);
    printf("%016x\n", 170);

    return EXIT_SUCCESS;
}
```

Would print:

```
aa
AA
0xaa
0XAA

beef
BEEF
0xbeef
0XBEEF

  aa
00aa
              aa
00000000000000aa
```

The Exercise

Create a program called `addition2.c`. This program should ask for two positive integers using the message `Please enter two positive integers:` then display the sum of the integers in hexadecimal as ***`n + n = sum`***.

> Make sure to replace the ***n*** with the numbers entered in the same order in hexadecimal and the ***sum*** with the sum of the two numbers in hexadecimal. Each hexadecimal number should be proceded by `0x` and should have two hexadecimal digits.

Some Examples

```
$ ./addition2
Please enter two positive integers: 2 5
0x02 + 0x05 = 0x07
```

```
$ ./addition2
Please enter two positive integers: 3 10
0x03 + 0x0a = 0x0d
```

```
$ ./addition2
Please enter two positive integers: 10 27
0x0a + 0x1b = 0x25
```

To run some simple automated tests:

```
$ 1511 autotest addition2
```

To run Styl-o-matic:

```
$ 1511 stylomatic addition2.c
Looks good!
```

You'll get advice if you need to make changes to your code.

Submit your work with the *give* command, like so:

```
$ give cs1511 wk03_addition2
```

Or, if you are working from home, upload the relevant file(s) to the `wk03_addition2` activity on Give Online.