# HW Problem 3: C Variable Bit Lengths

Wednesday, September 9, 2020 4:14 PM

(A) On the personal computer of your choice, write C code that declare three variables as unsigned short integers; then, set two of the variables to the values 5 and 6, respectively. Do the subtraction 5-6, putting the result in the third variable. Display that variable using the printf command so that you can explicitly state the format in which you would like the results printed. "%u" is used when "printf-ing" unsigned variables in decimal format. Then answer the following questions:

```c
#include <stdio.h>

int main(void)
{
    unsigned short var1;
    unsigned short var2;
    unsigned short var3;

    var1 = 5;
    var2 = 6;
    var3 = var1 - var2;

    printf("Result is: %u", var3);

    return 1;

}
```

(i) What value does the result have?

```
Result is: 65535
```

(ii) Given the result in (i), how many bits are used for an unsigned short integer? How do you know?

There are 16 bits used in unsigned short integers. You can tell because the result 6-5 will cause an overflow and give the maximum value available. The largest value for an unsigned representation is $2^n - 1$, with n being 16 that gives you $2^{16} - 1 = 65535$.

(B) Repeat (A) using unsigned integers instead of unsigned short integers

```
Result is: 4294967295
```

Given the result is larger and more difficult to recognize as a power of 2, the maximum value can be found by solving for $2^n - 1 = 4294967295$.
$\log_2(4294967295 + 1) = 32$.
This shows that unsigned integers use 32bits.

(C) Repeat (A) using unsigned long integers instead of unsigned short integers

```
Result is: 18446744073709551615
```

The same method can be used as part B.
$\log_2(18446744073709551615 + 1) = 64$
Unsigned long integers use 64bits.