

CMPE1250 – Understanding Bitwise Expressions and Types in C

Answer the following questions. **You must show all work** for any multi-step operations. Indicate any truncations, precedence*, and associativity* rules/events at the relevant steps. You may use a calculator to convert values, but a calculator may only be used to validate steps in every other circumstance.

// Q1

```
{
    unsigned int i = 5632 & 0x321 >> 3;
    // what is i in binary
    // what is i in hex
    // what is i in decimal
}
```

5632

0001	0110	0000	0000
0000	0000	0110	0100
<hr/>			
0000	0000	0000	0000

0x321

0000	0000	0011	0010	0001
<hr/>				
>> 3				

// Q2

```
{
    unsigned int i = (unsigned char)5632 | (0x321 >> 3);
    // what is i in binary
    // what is i in hex
    // what is i in decimal
}
```

5632

↓

(u) 0001 0110 (0000 0000)

(6x64) 0000 0000 0110 0100

↓

0000 0000 0110 0100

0x321

↓ 773

0000 0011 0010 0001

6 4

// Q3

```
{
    unsigned char i = (unsigned char)(5632 & (0xF371 >> 3));
    // what is i in binary
    // what is i in hex
    // what is i in decimal
}
```

5632

↓

0001 0110 0000 0000

0001 1110 0110 1110

↓

0001 0110 (0000 0000)

0xF371

↓ 773

0000 1111 0011 0111

1 E 6 E

// Q4

```
{
    unsigned int i = (unsigned char)(0x12345678ul / 4);
    // what is i in binary
    // what is i in hex
    // what is i in decimal
}
```

4 → shift 2
2²=4
0000 0000 1001 1110
0x9E
158
(0C) 0000 0010 0011 0100 0101 0110 0111 1000
0x9E

// Q5

```
{
    unsigned int i = (0b10101000100010001000100010001000 >> 2 & 0xF5A56832u);
    // what is i in binary
    // what is i in hex
    // what is i in decimal
}
```

0000 0000 0000 0010
2
0010 1010 1000 1001 0001 0100 1100 1010
1111 0101 1010 0101 0110 1000 0011 0010
0010 0000 1000 0001 (0000 0000 0000 0010)
as (01) 2 truncate

// Q6

```
{
    unsigned int i = ((5 * 0x34 + 0b1101010011) << 4 & (0b110011001100 >> 2));
    // what is i in binary
    // what is i in hex
    // what is i in decimal
}
```

③ 0011 0011 0011 0011 >> 2

① 0001 0100 0000
0x1A0

③ 0000 0011 0011 0011
& 0100 1111 0011 0000
0000 0011 0011 0000
0x0330

② 0001 1010 0000
+ 0011 0101 0011
0100 1111 0011 0000
④ 0100 1111 0011 0000
0x4C3