Study Questions: Set No. 11
Introduction to C
Tuesday December 3, 2013

Covering:

Chapter 20

1. Show the output produced by each of the following program fragments. Assume that `i`, `j`, and `k` are unsigned short variables.

```
a. i = 0xFF7F;
   printf("0x%x\n", i >> 1);

b. i = 0x0004;
   printf("0x%x\n", i << 3);

c. i = 0x0004;
   printf("0x%x\n", i << i);

d. i = 0xDFFF; j = 0xFF7F;
   printf("0x%x\n", i & j);

e. i = 0x2840; j = 0x0004;
   printf("0x%x\n", i | j);

f. i = 0xDFFF; j = 0x0004;
   printf("0x%x\n", i & ~j);

g. i = 0xDFFF; j = 0x2840;
   printf("0x%x\n", i ^ j);

h. i = 8; j = 9;
   printf("%d\n", i >> 1 + j >> 1);

i. i = 1;
   printf("%d\n", i & ~i);

j. i = 2; j = 1; k = 0;
   printf("%d\n", ~i & j ^ k);

k. i = 7; j = 8; k = 9;
   printf("%d\n", i ^ j & k);
```

2. Describe a simple way to *toggle* a bit (change it from 0 to 1 or from 1 to 0).
   Illustrate the technique by writing a statement that toggles bit 4 of the variable `i`.

3. Explain the effect of following function.

```
void f1(char *x, char *y)
{
  *x ^= *y;
  *y ^= *x;
  *x ^= *y;
}
```

Copyright © 2013 Mahmoud El-Sakka

4. In computer graphics, colors are often stored as three numbers, representing red, green, and blue intensities. Suppose that each number requires eight bits, and we'd like to store all three values in a single long integer.

   a. Write a macro named `MK_COLOR` with three parameters (the red, green, and blue intensities). `MK_COLOR` should return a `long` in which the last three bytes contain the red, green, and blue intensities, with the red value as the last byte and the green value as the next-to-last byte.

   b. Write macros named `GET_RED`, `GET_GREEN`, and `GET_BLUE` that, when given a color as an argument, see part (a), it returns its 8-bit red, green, and blue intensities.

5. Use the bitwise operators to write the following function:

```
unsigned short swap_bytes(unsigned short i);
```

`swap_bytes` should return the number that results from swapping the two bytes in `i`. (Short integers occupy two bytes on most computers.) For example, if `i` has the value `0x1234` (`00010010 00110100` in binary), then `swap_bytes` should return `0x3412` (`00110100 00010010` in binary). Test your function by writing a program that reads a number in hexadecimal, then writes the number with its bytes swapped:

```
Enter a hexadecimal number (up to four digits): 1234
Number with bytes swapped: 3412
```

Hint: Use the `%hx` conversion to read and write the hex numbers.
Condense the `swap_bytes` function so that its body is a single statement.

6. Write the following functions:

```
unsigned int rotate_left(unsigned int i, int n);
unsigned int rotate_right(unsigned int i, int n);
```

`rotate_left` should return the result of shifting the bits in `i` to the left by `n` places, with the bits that were *shifted off* moved to the right end of `i`.
For example, the call `rotate_left(0x12345678, 4)` should return `0x23456781` if integers are 32 bits long.
`rotate_right` is similar, but it should *rotate* bits to the right instead of the left.

7. Let `f` be the following function:

```
unsigned int f(unsigned int i, int m, int n )
{
  return     ( i >> (m + 1    - n)) & ~(~0 << n);
}
```

   a. What is the value of `~(~0 << n)`?

   b. What does this function do?

8. Write the following function:

```
int count_ones(unsigned char ch);
```

`count_ones` should return the number of 1 bits in `ch`.

9. Write the following function:

```
unsigned int reverse_bits(unsigned int n);
```

`reverse_bits` should return an unsigned integer whose bits are the same as those in `n` but in reverse order.

Copyright © 2013 Mahmoud El-Sakka

10. Each of the following macros defines the position of a single bit within an integer:

```
#define SHIFT_BIT 1
#define CTRL_BIT  2
#define ALT_BIT   4
```

The following statement is supposed to test whether any of the three bits have been set, but it never displays the specified message. Explain why the statement doesn't work and show how to fix it. Assume that `key_code` is an `int` variable.

```
if(key_code & (SHIFT_BIT | CTRL_BIT | ALT_BIT) == 0)
  printf("No modifier keys pressed\n");
```

11. The following function supposedly combines two bytes to form an unsigned short Integer. Explain why the function doesn't work and show how to fix it.

```
unsigned short create_short(unsigned char high_byte,
                            unsigned char low_byte)
{
  return high_byte << 8 + low_byte;
}
```

12. If `n` is an `unsigned int` variable, what effect does the following statement have on the bits in `n`?

```
n &= n - 1;
```

13. When stored according to the *IEEE floating-point standard*, a `float` value consists of a 1-bit sign (the leftmost bit, i.e., most significant bit), an 8-bit exponent, and a 23-bit fraction, in that order. Design a structure type that occupies 32 bits, with bit-field members corresponding to the sign, exponent, and fraction. Declare the bit-fields to have type `unsigned int`.

14. Assume that the variable `s` has been declared as follows:

```
struct
{
  int flag: 1;
} s;
```

With some compilers, executing the following statements causes 1 to be displayed, but with other compilers, the output is -1. Explain the reason for this behaviour.

```
s.flag = 1;
printf("%d\n", s.flag);
```

How can this problem be avoided?

Copyright © 2013 Mahmoud El-Sakka