

This daily will allow you to practice more with the bit wise operators and shifts. Consider the following modification of the main program from daily 3:

```
#include <stdio.h>

void set_flag(unsigned int* flag_holder, int flag_position);
void unset_flag(unsigned int * flag_holder, int flag_position);
int check_flag(unsigned int flag_holder, int flag_position);
void display_32_flags(unsigned int flag_holder);

int main(int argc, char* argv[])
{
    unsigned int flag_holder = 0;

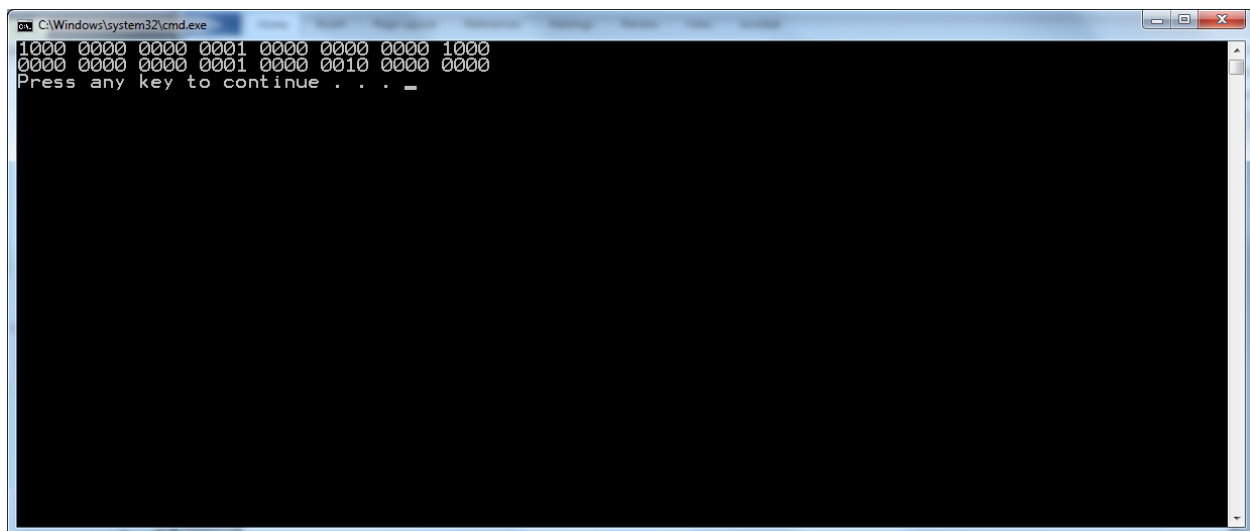
    set_flag(&flag_holder, 3);
    set_flag(&flag_holder, 16);
    set_flag(&flag_holder, 31);

    display_32_flags(flag_holder);

    unset_flag(&flag_holder, 31);
    unset_flag(&flag_holder, 3);
    set_flag(&flag_holder, 9);

    display_32_flags(flag_holder);
    return 0;
}
```

Write the code for the definition of `unset_flag` and `display_32_flags` so that the output of your program looks like the following:



```
C:\Windows\system32\cmd.exe
1000 0000 0000 0001 0000 0000 0000 1000
0000 0000 0000 0001 0000 0010 0000 0000
Press any key to continue . . . _
```

You can think of the `unset_flag` function as taking an integer and making sure that the n^{th} bit is a 0. You may find the `~` operator useful. It is used to “flip the bits” of a number making all the zero values 1’s and all the 1’s zeroes. As in the previous daily, the shifting operators and the bitwise and (`&`) and or (`|`) may also be useful. If you are doing multiplication or division then you are doing it wrong. The `display_32_flags` function should just print the information to the screen as was given in the previous assignment (just turn it into a function instead).

At the top of your code you should have a comment section that has the following format:

```
/******  
    Author: <your name>  
    Date: <Today's date>  
    Effort: <Time you spent on this project>  
    Purpose: <Purpose of this assignment in your own words>  
******/
```