COMP500 / ENSE501: Week 11 – Exercise:

EXERCISE NAME: *Roulette Comments*

A Roulette table is set out as follows (note the 2:1 columns are shown horizontally):

0	3	6	9	12	15	18	21	24	27	30	33	36	2:1
	2	5	8	11	14	17	20	23	26	29	32	35	2:1
	1	4	7	10	13	16	19	22	25	28	31	34	2:1
	1st 12				2nd 12				3rd 12				
	1 to 18		EVEN		RED		BLACK		ODD		19 to 36		

A Roulette wheel contains numbers between 0 and 36, inclusive. Write a modular program that spins a virtual Roulette wheel, then analyses the resulting number to categorise it as follows: red/black, odd/even, 1st/2nd/3rd 12, "1 to 18"/"19 to 36", and 1st/2nd/3rd 2:1 column.

For each function in this exercise, write a function comment in the following style:

```
/*
 * Function: compute_factorial
 * ----------------------------
 *   Returns the factorial of the value input
 *
 *   input: the number to compute the factorial of
 *
 *   returns: the factorial of input
 *
 *   pre: input must be positive.
 *
 *   post: value returned will be greater than zero.
 */
int compute_factorial(int input)
{
    // Insert function code here...
}
```

Ensure you also carefully utilise any pre- or post-conditions for each function created, and document the requirements in the function's function comment.

Declare and define a function named **`print_colour`**, that takes in one parameter, a number, and then prints out in the following style, based upon the number input:

```
... 13 is black,
```

Declare and define a function named **`print_odd_or_even`**, that takes in one parameter, a number, and then prints out in the following style, based upon the number input:

```
... 13 is odd,
```

Declare and define a function named **print_twelve_range**, that takes in one parameter, a number, and then prints out in the following style, based upon the number input:

```
... 13 is in the 2nd 12,
```

Declare and define a function named **print_low_or_high**, that takes in one parameter, a number, and then prints out in the following style, based upon the number input:

```
... 13 is in 1 to 18,
```

Declare and define a function named **print_column**, that takes in one parameter, a number, and then prints out in the following style, based upon the number input:

```
... 13 is in the first 2:1 column.
```

Declare and define a function named **prompt_for_spin** which prints out, and takes in user input, in the following style:

```
Spin the wheel (y/n)? y
```

The **prompt_for_spin** function must return the character input by the user to the caller.
Declare and define a function named **get_wheel_spin** which returns a random number between 0 and 36. This function must have no side effects, and hence only return the value without printing anything.

In the **main** function, call each of the previously declared functions to create a program which achieves the overall result as follows:

```
Spin the wheel (y/n)? y

The wheel shows 13
... 13 is black,
... 13 is odd,
... 13 is in the 2nd 12,
... 13 is in 1 to 18,
... 13 is in the first 2:1 column.

Spin the wheel (y/n)? y

The wheel shows 0
... 0 is green.

Spin the wheel (y/n)? n
```

In addition, create any other modular functions that help realise the overall game play algorithm.

Ensure the program output is exactly as described, and that the whitespace of your source code is well formatted. Utilise good naming practices when declaring variables.

Test your program with a variety of input to ensure the implementation is robust.