Favourite Number 3

> This is a **pair** exercise and must be competed in your **tutorial** or **lab** with your partner.

In the warmups for the last two weeks you have looked at my favourite number, 17, and the user's favourite number. This week you are going to write some functions involving loops about the user's favourite number.

Copy the program `favourite3.c` from the course account to your directory by typing **(make sure you type the dot at the end)**:

```
$ cp /web/cs1511/17s2/week04/files/favourite3.c .
```

The dot '.' is a shorthand for the current directory; note that **there is a space between `favourite3.c` and the next dot**.

You can check that the file has been copied by typing:

```
$ ls
favourite3.c
```

This file already has a main function that is complete, do not change anything in it.

Functions to Complete

There are two function prototypes for you to implement inside `favourite3.c`. Open `favourite3.c` and complete the functions.

These are:

```
void printTimesTable (int n);
int factorial (int n);
int isPerfectNumber (int n);
```

The `void printTimesTable (int n)` function takes in an integer, `n`, and prints out its "times table" up to 12 in this format (taking 17 as an example):

```
1 x 17 = 17
2 x 17 = 34
3 x 17 = 51
4 x 17 = 68
5 x 17 = 85
6 x 17 = 102
7 x 17 = 119
8 x 17 = 136
9 x 17 = 153
10 x 17 = 170
11 x 17 = 187
12 x 17 = 204
```

You should be using the `#define` 'd value of LAST_TIMES_TABLE.

Both the `factorial` and `isPerfectNumber` functions return a value.

The `int factorial (int n)` function takes in a positive integer, `n` , and should return `n` factorial (i.e. n x (n-1) x (n-2) x ... x 3 x 2 x 1).

The `int isPerfectNumber (int n)` function takes in a positive integer and returns 1 if it is a perfect number, and 0 otherwise.

A perfect number is a number whose factors (not including itself) sum up to the number. For example, 6 is a perfect number because 1 + 2 + 3 = 6. You should be using the `#define` 'd values of TRUE and FALSE.

Some examples

```
$ ./favourite3
Enter your favourite number: 4
You entered 4.
4! = 24
1 x 4 = 4
2 x 4 = 8
3 x 4 = 12
4 x 4 = 16
5 x 4 = 20
6 x 4 = 24
7 x 4 = 28
8 x 4 = 32
9 x 4 = 36
10 x 4 = 40
11 x 4 = 44
12 x 4 = 48
4 is not a perfect number.
```

```
$ ./favourite3
Enter your favourite number: 6
You entered 6.
6! = 720
1 x 6 = 6
2 x 6 = 12
3 x 6 = 18
4 x 6 = 24
5 x 6 = 30
6 x 6 = 36
7 x 6 = 42
8 x 6 = 48
9 x 6 = 54
10 x 6 = 60
11 x 6 = 66
12 x 6 = 72
6 is a perfect number.
```

```
$ ./favourite3
Enter your favourite number: -1
You didn't enter a positive number.
```

To run some simple automated tests:

```
$ 1511 autotest favourite3
```

To run Styl-o-matic:

```
$ 1511 stylomatic favourite3.c
Looks good!
```

You'll get advice if you need to make changes to your code.

Submit your work with the *give* command, like so:

```
$ give cs1511 wk04_favourite3
```

Or, if you are working from home, upload the relevant file(s) to the `wk04_favourite3` activity on Give Online.