Submit your source code is to be uploaded to Canvas using `YourName_Assignment5.c` format.

Your functions must have a prototype!
Use simple short comments to only when needed.
Use indentations to make your code visibly clear and easy to follow.
Make the output display of your program visually appealing.
You MUST compile your code and test is multiple times before submission. Codes that can't be compiled would be graded as zero.

There is 10 points deduction for not following proper submission structure.

For any integer *n* > 0, *n*`!(n factorial)` is defined as the product

n * *n* - 1 * *n* - 2 … * 2 * 1.

And `0!` is defined to be `1`.

It is sometimes useful to have a closed-form definition instead; for this purpose, an approximation can be used. R.W. Gosper proposed the following approximation formula:

$$n! \approx n^n e^{-n} \sqrt{\left(2n + \frac{1}{3}\right)\pi}$$

a) Create a function takes `n` as input and returns the approximation for factorial value back.

b) Create another function takes n as input and computes then returns the accurate value for *n!=* *n* * *n* - 1 * *n* - 2 … * 2 * 1.

c) Your program should prompt the user to enter an integer `n`, call both functions to compute the approximate and accurate values for `n!` and then display the results. The message displaying the result should look something like this:

```
5! equals approximately 119.97003
5! is 120 accurately.
```

d) Test the program on nonnegative integers less than 10. (A type `int` might not accommodate overly large numbers). Find the difference between the two results for accurateness, then compute the percent error. Is the approximation a good representation of the actual value? Use printf to display the error.

$$percent\ error = \frac{accurate\ value - approximate\ value}{accurate\ value} \times 100$$

TIP: Be careful with the type conversions. Be sure to use a named constant for PI, and use the approximation `3.14159265`.

Each section is worth 25 points for a total of 100 points.