CS 159 – Homework #3
Due: Monday February 21 at 11:00pm (time local to West Lafayette, IN).
10 Points Possible

Problem: A manufacturing facility is evaluating their current processes by which their primary commercial device is produced. Given the percentage of defective units produced determine the probability that there will R defective units from a sample size of N. In addition the program should calculate the probability that there will be no defective units from that sample.

More background information on the mathematics needed to solve this type of problem can be found in many on-line sources including: https://en.wikipedia.org/wiki/Binomial_distribution

- Use the estimate provided (formula on right) for factorial where needed in your solution.

$$x! = (x^x e^{-x}) * \sqrt{(2x + \frac{1}{3})\pi}$$

Example Execution #1:

```
Enter percentage of defective units produced -> 1.75
Enter number of units to sample -> 100
Enter number of defective units in sample -> 5

Average number of defective units: 1.75%
Selecting 100 units with 0 being defective has a probability of: 0.167
Selecting 100 units with 5 being defective has a probability of: 0.023
```

Example Execution #2:

```
Enter percentage of defective units produced -> 18
Enter number of units to sample -> 10
Enter number of defective units in sample -> 4

Average number of defective units: 18.00%
Selecting 10 units with 0 being defective has a probability of: 0.134
Selecting 10 units with 4 being defective has a probability of: 0.067
```

Example Execution #3:

```
Enter percentage of defective units produced -> 10
Enter number of units to sample -> 50
Enter number of defective units in sample -> 6

Average number of defective units: 10.00%
Selecting 50 units with 0 being defective has a probability of: 0.005
Selecting 50 units with 6 being defective has a probability of: 0.154
```

Example Execution #4:

```
Enter percentage of defective units produced -> 15.5
Enter number of units to sample -> 30
Enter number of defective units in sample -> 4

Average number of defective units: 15.50%
Selecting 30 units with 0 being defective has a probability of: 0.006
Selecting 30 units with 4 being defective has a probability of: 0.198
```

All course programming and documentation standards are in effect for this and each assignment this semester. Please review this document!

Academic Integrity Reminder: Please review the policies of the course as they relate to academic integrity. The assignment you submit should be your own original work. You are to be consulting only course staff regarding your specific algorithm for assistance. Collaboration is not permitted on individual homework assignments.

Additional Requirements:

1. Add the homework assignment header file to the top of your program. A description of your program will need to be included in the assignment header. This particular header can be added to your file by entering `:hhw` while in command mode in `vi`.
2. **Each of the example executions provided for your reference represents a single execution of the program.** Your program must accept input and produce output **exactly** as demonstrated in the example executions, do not add any "bonus" features not demonstrated in the example executions. Your program will be tested with the data seen in the example executions and an unknown number of additional tests making use of meaningful data.
    - The percentage of defective units may be input as a floating-point (`double`) value. The remaining user input will be integer data. All input values will be greater than zero.
    - The use of the `math.h` library is expected.
3. For this assignment you will be **required** to implement the user-defined functions (from chapter 4). Failing to follow course standards as they relate to good user-defined function use will result in a **zero for this assignment.**
4. Revisit **course standards as it relates what makes for good use of user-defined functions, what is acceptable to retain in the `main` function, and when passing parameters by address is appropriate.**
    - In many cases user-defined function use should result in a `main` function that only declares variables and makes function calls.
5. Course standards **prohibit** the use of programming concepts not yet introduced in lecture. For this assignment you can consider all material in the **first four chapters** of the book, notes, and lectures to be acceptable for use.
    - Course standards **prohibit** the use of programming concepts beyond the material found in the first four chapters of the book, notes, and lectures.
6. A program **MUST** compile, be submitted through Vocareum as demonstrated during the lab #0 exercise, and successfully submitted prior to the posted due date to be considered for credit. The C-file you submit must be named exactly: `hw03.c`, no variation is permitted.

Course Programming and Documentation Standards Reminders:

- Use the course function header (`vi` shortcut `:hfx` while in command mode) for every user-defined function in your program.
    - List and comment **all parameters** to a function, one per line, in the course function header.
    - **All function declarations** will appear in the global declaration section of your program.
    - **The user-defined function definitions will appear in your program after the `main` function.**
- Indent all code found within the `main` function **exactly** two spaces.
- Place a **single space** between all operators and operands.
- Comment **all** variables to the right of each declaration. Declare only one variable per line.
- Notice that several programs (see program 2-9 on pages 74-75) in the programming text use a single line comment to indicate the start of the local declaration and executable statement sections of the `main` function.
    - At no point during the semester should these two sections ever overlap.
- Select **meaningful identifiers** (names) for all variables in your program.
- Do not single (or double) space the entire program, **use blank lines when appropriate**.
- There is no need to include example output with your submission.

When you submit... only the final successful submission is kept for grading. All other submissions are over-written and cannot be recovered. You may make multiple submissions but only the last attempt is retained and graded.

- Verify in the confirmation e-mail sent to you by the course that you have submitted the correct file to the correct assignment.
- Leave time prior to the due date to seek assistance should you experience difficulties completing or submitting this assignment. All attempts to submit via a method other than through the appropriate assignment on Vocareum will be denied consideration.

Assignment deadlines... are firm and the electronic submission will disable promptly as advertised. We can only grade what you are able submit via Vocareum prior to the assignment deadline.