Assignment 01

CSE24101

March 12, 2022

1 Project structure

Submit your project as a .zip file, consisting of 5 files:

main.cpp — contains a `main` method which reads in k and prints out the table listed below

check.cpp — contains a `main` method and does unit testing (see below)

functions.cpp — contains a `factorial` function and a `probability` function

functions.h — contains prototypes for `factorial` and `probability`

Makefile — must have rules to build both `main` and `check`

Each source code file must include a comment at the top with a short description and your name.

2 Calculations

Create a C++ program which reads in a positive integer k from the user. It must repeatedly read in a positive integer, and exit if a negative integer or something that cannot be parsed as an integer is encountered. Then, it will print out a table of probabilities, for all numbers n from 1 to k, how likely it is that any two people in a random sample of n will share the same birth month, if there are k months in a year.

The formula for calculating this is:

$$p(n) = 1 - \frac{k!}{k^n(k-n)!}$$

You must choose data types large enough to handle $k = 12$, at least. To calculate exponentials, use the `std::pow` function (you will need to include `cmath`). Note: k^n is probably too large to fit in an integer type (even `long long`), so use a `double` for the result of that.

Make one function for calculating factorial, and one function for calculating the probability. It is up to you to decide the best return types and parameters/parameter types.

3 Two executables

You will produce *two* executable files. That means that you will have *two* `main` function. Sounds crazy, right?

One of the executable files (with `main.cpp`) will read in input from the user and output a nice-looking table.

The other executable file (with `check.cpp`) will only do *unit testing*. It will not read in any input and will not produce much output, but is for automated use.

Your Makefile will have to have 2 rules, one for each executable. Both executables will depend on `functions.o`

3.1 main.cpp

Make one `main` function in your `main.cpp` file.

Read in an integer, `k`, from the user. If the input is invalid, exit right away (you don't need to ask again). Then, make a `for` loop to go through all values n, from 1 to k. You must print out each probability to 7 digits after the decimal point.

To be clear, you must read in *one* number (k). Then you set up a `for` loop with the variable n, calling your probability function with all values of n from 1 to k.

The values of n and the probabilities must each be aligned and the output must look *exactly* like the following for $k = 12$, where the first column is n and the second column is the probability:

```
 1   0.0000000
 2   8.3333333
 3  23.6111111
 4  42.7083333
 5  61.8055556
 6  77.7199074
 7  88.8599537
 8  95.3583140
 9  98.4527713
10  99.6131928
11  99.9355321
12  99.9946277
```

Note that since the probabilities are percentages, they should be multiplied by 100.

3.2 check.cpp

Make the other `main` function in your `check.cpp` file.

It will contain *unit tests*. It must call the `factorial` and `probability` functions and ensure they are returning the correct values. If there are any incorrect values discovered in testing, you must print an error message. If there are no incorrect values discovered in testing, you should print only a single line of output that says "All tests passed".

You must test the `factorial` function with at least 3 different values and ensure that it returns the numbers you expect.

You must test the `probability` function at least 5 times, including at least 2 different values of `k`. Because the `probability` function returns a `double` and `double`s cannot be reliably tested for equality, you may set up your checks such that the value returned is *close* (e.g., within 0.0001) of the expected value.