John-Craig Borman

FE 522

Assignment 1 – Part 1

3/12/19

Problem 1:

a. I wrote two functions. First, `sum_of_squares(n)` that recursively calculates the sum of grains in all squares from 1 ... n. Second, `num_squares(grains)` that loops over an if-statement checking if the total number of grains up to square `i` is greater than the number of `grains` passed as an argument. This function returns the number (int) of squares necessary to produce the desired number of `grains`.
b. When the number gets too large to be represented by an int, that integer variable becomes overloaded and turns into the maximum negative value allowed by an integer. Why? Because the first bit of the integer that is reserved for sign becomes overloaded by the number that is too large and therefore becomes negative. For the double, the double looses precision and becomes only an approximate representation of the exact number.
   - The max number of grains that an integer can represent is 29 squares worth.

```
The largest number of grains stored in an int are (29 squares):
    As an integer: 1073741823
```

   - Doubles will calculate the approximate number of grains up until: 1.79769e+308

Problem 2:

a. Please view the test cases in the appropriate file. Bugs were fixed and the factorial functionality was added.

Problem 3:

- For this problem I decided to create a template RandomVector class that can be defined for any numeric type (e.g. ints and doubles). This was so that I did not need to create redundant vectors or functions for discrete/continuous Random Variables. Further, I created 6 functions to sample different distributions that make calls to a general `sample_dist` function that returns a vector of `n` samples of the given distribution `d`.
- In `main` I create six instances of RandomVector, from 3 discrete and 3 continuous distributions. I then add their moments as vectors to another vector (2D vector of means, variances) and finally write these to a .CSV located in /output/dist-moments.csv.