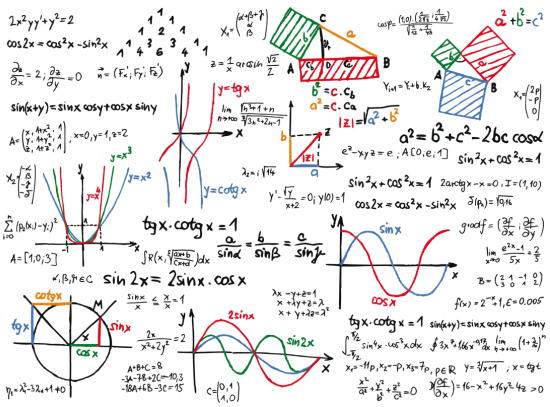
B4 - Mathematics

B-MAT-400

203hotline

Avoiding Overload



203hotline

binary name: 203hotline
language: everything working on "the dump"
compilation: when necessary, via Makefile, including re, clean and fclean rules

- The totality of your source files, except all useless files (binary, temp files, obj files,...), must be included in your delivery.
- All the bonus files (including a potential specific Makefile) should be in a directory named *bonus*.
- Error messages have to be written on the error output, and the program should then exit with the 84 error code (0 if there is no error).

A 25-phone hotline owner reckons 3500 people could possibly call during each 8-hour day, and would like to know the probability of an overload (that is, the probability of no line being available), depending on the average duration of calls.

The random variable representing the number of people calling at a given time follows the binomial distribution, with calls being independent from each other. You're also thinking about estimating the binomial ditribution with a Poisson distribution, so it can be used on a larger scale.

Your first task is to compute the binomial coefficient $\binom{n}{k}$ given k and n (emphasizing the computation speed and stack optimization).

Your second task is to compare the binomial and Poisson distributions, given the average duration of calls, by printing:

- the probabilities of getting n simultaneous calls (for n increasing from 0 to 50),
- the probability of an overload,
- the computation time.

USAGE

Terminal

```
~/B-MAT-400> ./203hotline -h
USAGE
    ./203hotline [n k | d]

DESCRIPTION
    n       n value for the computation of C(n, k)
    k       k value for the computation of C(n, k)
    d       average duration of calls (in seconds)
```

EXAMPLES

Your program output has to be strictly identical to the one below.

Terminal

```
~/B-MAT-400> ./203hotline 100 18
18-combinations of a set of size 100:
30664510802988208300
```

Terminal

```
~/B-MAT-400> ./203hotline 180
Binomial distribution:
0 -> 0.000      1 -> 0.000      2 -> 0.000      3 -> 0.000      4 -> 0.000
5 -> 0.000      6 -> 0.000      7 -> 0.000      8 -> 0.000      9 -> 0.001
10 -> 0.002     11 -> 0.004     12 -> 0.008     13 -> 0.013     14 -> 0.021
15 -> 0.030     16 -> 0.041     17 -> 0.053     18 -> 0.065     19 -> 0.075
20 -> 0.082     21 -> 0.085     22 -> 0.085     23 -> 0.081     24 -> 0.074
25 -> 0.064     26 -> 0.054     27 -> 0.044     28 -> 0.034     29 -> 0.026
30 -> 0.019     31 -> 0.013     32 -> 0.009     33 -> 0.006     34 -> 0.004
35 -> 0.002     36 -> 0.001     37 -> 0.001     38 -> 0.000     39 -> 0.000
40 -> 0.000     41 -> 0.000     42 -> 0.000     43 -> 0.000     44 -> 0.000
45 -> 0.000     46 -> 0.000     47 -> 0.000     48 -> 0.000     49 -> 0.000
50 -> 0.000
Overload: 21.4%
Computation time: 1.71 ms

Poisson distribution:
0 -> 0.000      1 -> 0.000      2 -> 0.000      3 -> 0.000      4 -> 0.000
5 -> 0.000      6 -> 0.000      7 -> 0.000      8 -> 0.000      9 -> 0.001
10 -> 0.002     11 -> 0.004     12 -> 0.008     13 -> 0.013     14 -> 0.021
15 -> 0.030     16 -> 0.042     17 -> 0.053     18 -> 0.065     19 -> 0.075
20 -> 0.082     21 -> 0.085     22 -> 0.085     23 -> 0.081     24 -> 0.073
25 -> 0.064     26 -> 0.054     27 -> 0.044     28 -> 0.034     29 -> 0.026
30 -> 0.019     31 -> 0.013     32 -> 0.009     33 -> 0.006     34 -> 0.004
35 -> 0.002     36 -> 0.001     37 -> 0.001     38 -> 0.001     39 -> 0.000
40 -> 0.000     41 -> 0.000     42 -> 0.000     43 -> 0.000     44 -> 0.000
45 -> 0.000     46 -> 0.000     47 -> 0.000     48 -> 0.000     49 -> 0.000
50 -> 0.000
Overload: 21.5%
Computation time: 0.34 ms
```