Exercise 1

Write an interface `RandomVariableInterface` with at least the following methods:

- `double generate()`, which returns one realization of the random variable;
- `double getAnalyticMean()`, which returns the analytic expectation of the random variable;
- `double getAnalyticStdDeviation()`, which returns the analytic standard deviation of the random variable;
- `double getSampleMean(int n)`, which returns the mean of n independent realizations of the random variable;
- `double getSampleStdDeviation(int n)`, which returns the standard deviation of n independent realizations of the random variable;
- `double getDensityFunction(double x)`, which returns the density function of the random variable, evaluated at x;
- `double getCumulativeDistributionFunction(double x)`, which returns the distribution function of the random variable, evaluated at x;
- `double getQuantileFunction(double x)`, which returns the quantile function of the random variable, evaluated at x.

Write then an abstract class `RandomVariable` implementing `RandomVariableInterface`, which gives the implementation of all the methods of `RandomVariableInterface` whose implementation does not depend on the specific distribution of the random variable into consideration.

Exercise 2

Write the following two classes extending `RandomVariable`:

- `ExponentialRandomVariable`, which generates exponentially distributed random variables of parameter λ by inversion sampling;
- `NormalRandomVariable`, which generates a normal $\mathcal{N}(\mu, \sigma^2)$ by inversion sampling. The method `quantileFunction(double x)` must return the approximated quantile function computed using the approximation
$$q_x \approx -t + \frac{c_0 + c_1 t + c_2 t^2}{1 + d_1 t + d_2 t^2 + d_3 t^3}, \tag{1}$$
where $t = \sqrt{\log\left(\frac{1}{x^2}\right)}$, for $x \leq 0.5$, $c_0 = 2.515517$, $c_1 = 0.802853$, $c_2 = 0.010328$, $d_1 = 1.432788$, $d_2 = 0.189269$, $d_3 = 0.001308$, see Abramovitz and Stegun's book *Handbook of mathematical functions* (you can find it also online). You then have to use (1) together with properties of the normal distribution to get the approximation also for $x > 0.5$.

  The method `cumulativeDistributionFunction(double x)` must instead return the cumulative distribution function as
$$\text{cdf}(x) = 0.5\left(1 + \text{erf}\left(\frac{x-\mu}{\sqrt{2\sigma^2}}\right)\right),$$
where $\text{erf}(y)$ is approximated using the expansion
$$\text{erf}(y) = \frac{2}{\sqrt{\pi}} \sum_{n=0}^{\infty} \frac{(-1)^n y^{2n+1}}{n!(2n+1)}.$$
The order of the expansion is a field of the class.

Test your classes for exponential and normal random variables of your choice: for example, you can compute mean and sample mean, standard deviation and sample standard deviation, and see if the values converge when you increase the number of realizations. Another test you can do is to compute the approximated quantile and distribution function. For example, the quantile of a probability of 0.95 of a standard normal random variable should be equal to 1.645211440143815.

Exercise 3

The package `com.andreamazzon.exercise6.confidenceintervals` deals with the computation of confidence intervals for the mean of a sample of given length of independent realizations of a given random variable. You can find most part of the code already in the repository. You can see that the classes of the previous exercise are used, so you will get an error if you don't write and import the classes you need. In particular, there is an abstract class `MeanConfidenceIntegral` containing two abstract methods

`getLowerBoundConfidenceInterval(double level)`

and

`getUpperBoundConfidenceInterval(double level)`

which compute the lower bound and the upper bound, respectively, of the confidence interval at the given level for the mean of a sequence of i.i.d. random variables with a given distribution.

This two abstract methods are implemented in the two derived classes `ChebyshevConfidenceIntegral` and `CLTConfidenceIntegral` where lower and upper bounds are calculated using Chebyshev's inequality and the Central Limit theorem, respectively.

(a) Prove that the implementation of the methods in the two above classes derives indeed from the Chebyshev's inequality and from the Central Limit Theorem, as they are stated in the script (session 03.pdf).

(b) Write the implementation of the concrete method `frequenceOfInterval(int numberOfMeanComputations, double level)` in `MeanConfidenceIntegral`, that computes the frequency with which the mean of the sample falls inside the confidence interval for the given confidence level `level`, when the mean is computed `numberOfMeanComputations` times.

(c) Test the classes by running the class `ConfidenceIntervalsTesting`. Try to understand the implementation of all the classes of the package.