Exercise 1

Look at the class `DiscrepancyOneDimension` in `com.andreamazzon.exercise5.discrepancy`. Here you have to implement two `public static double` methods, returning the discrepancy and the star discrepancy, respectively, of a set $\{x_1, \dots, x_n\}$ of one-dimensional points. You can follow the hints given in the `Javadoc` documentation. In particular, the discrepancy may be computed as

$$D(\{x_1, \dots, x_n\}) = \max_{a \in \{0, x_1, \dots, x_n\}} \max_{b \in \{x_1, \dots, x_n, 1\}, b > a} \max\left(b - a - \frac{|x_i \in (a,b)|}{n}, \frac{|x_i \in [a,b]|}{n} - (b-a)\right). \quad (1)$$

One can then use representation (1) by first writing a method that computes

$$\max_{b \in \{x_1, \dots, x_n, 1\}, b > a} \max\left(b - a - \frac{|x_i \in (a,b)|}{n}, \frac{|x_i \in [a,b]|}{n} - (b-a)\right) \quad (2)$$

for $a \in \{x_1, \dots, x_n\}$ fixed, and then compute the discrepancy as the maximum between the star discrepancy, which is

$$\max_{b \in \{x_1, \dots, x_n, 1\}, b > a} \max\left(b - \frac{|x_i \in (0,b)|}{n}, \frac{|x_i \in [0,b]|}{n} - b\right),$$

and the maximum of the values of (2). The method computing (2) is `getMaximumValue(double[] set, int position)` where `position` is the position of a in $\{x_1, \dots, x_n\}$ starting from position 0, and must be implemented.

However, of course you can avoid using the hint given above if you have other ideas.

Exercise 2

The method `getVanDerCorputSequence(int n, int base)` which has been now added to the class `com.andreamazzon.exercise4.VanDerCorputSequence` returns the first `n` elements of the Van der Corput sequence of base `base`, as a one-dimensional array. Use this method, and the methods implemented above, to implement the methods `getVanDerCorputStarDiscrepancy(int sequenceLength, int base)` and `getVanDerCorputDiscrepancy(int sequenceLength, int base)` in the class `VanDerCorputDiscrepancy` that you find in `com.andreamazzon.exercise5.discrepancy`. That is, you have to compute the star discrepancy and the discrepancy, respectively, of a Van der Corput sequence of given length and base.

Once we are able to compute the discrepancy of Van der Corput sequences with the methods above, the methods `plotVanDerCorputStarDiscrepancy(int maxSequenceLength, int base)` and `plotVanDerCorputDiscrepancy(int maxSequenceLength, int base)` plot the star discrepancy and the discrepancy of Van der Corput sequences for increasing dimensions. Here you only have to complete the definition of the `DoubleUnaryOperator starDiscrepancyFunction` and `starDiscrepancyFunction`, respectively.

Running the class `mainClass`, you can see the results. You should get for the first set discrepancy 0.375 and star discrepancy 0.25, and for the second set discrepancy 0.585 and star discrepancy 0.385.

Exercise 3

The class `com.andreamazzon.exercise4.MonteCarloIntegrationTwoDimensions` implements the Monte-Carlo approximation of

$$\int_0^1 \int_0^1 f(x,y)dxdy,$$

where $f : [0,1] \times [0,1] \to \mathbb{R}$. The integrand f is here represented by a field of type `BiFunction<Double, Double, Double>`. Complete the definition of the class by writing the constructor and giving the implementation of the method `computeIntegral()`.

In the same package, in the class `MonteCarloPiFromTwoDimensionsIntegration` we want to give an implementation of the approximation of π from the Monte-Carlo approximation of the area of the unit circle in $\mathbb{R}^2$, alternative to the one we have seen in the last weeks. Specifically, we want to use the implementation of the class above. In particular, the implementation of the method `generateMonteCarloComputations()` must be delegated to an object of type `MonteCarloIntegrationTwoDimensions`. Do this and complete the implementation of the constructor following the hint in the comments of the class.