Solution to exercise 3

We want to compute the Monte-Carlo approximation of an integral

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

where $f : (0,1) \times (0,1) \to \mathbb{R}$. In order to do so, we have to generate a given number N of random realizations $(x_i, y_i) \in (0,1) \times (0,1)$, $i = 1, \dots, N$, so that we can approximate

$$\int_0^1 \int_0^1 f(x,y)dxdy \approx \frac{1}{N} \sum_{i=1}^{N} f(x_i, y_i).$$

Of course, in order for the approximation to be good enough, the two realizations (x_i, y_i) have to be uncorrelated, for any $i = 1, \dots, N$. The intuition for this requirement is that, ideally, they have to fill all the set $(0,1) \times (0,1)$ when N tends to infinity, and not only specific regions.

With this in mind, let's now look at the two questions of the exercise.

(a) Suppose we have a class implementing `UniformRandomNumberSequence`, and specifically the method

```
double[] getSequenceOfRandomNumbers(),
```

by a Van der Corput sequence with a given base. That is, suppose that `getSequenceOfRandomNumbers()` returns a Van der Corput sequence. The question is then if we can use such an array to get the numbers $(x_i, y_i) \in (0,1) \times (0,1)$, for $i = 1, \dots, N$. If the function f we want to integrate was one-dimensional, this would had been fine. Indeed, for increasing N, the numbers produced by the sequence fill the interval $(0,1)$ in an uniform way.

However, things are not fine when coming to the two-dimensional case. Indeed, the Van der Corput sequence shows some serial dependence between its elements. For example, we have that

$$\left(x_{i+1} - \frac{1}{2}\right)\left(x_i - \frac{1}{2}\right) < 0$$

for $i > 2$. For this reason, the requirement that the two numbers (x_i, y_i) are uncorrelated for any $i = 1, \dots, N$ is violated.

So, we cannot achieve a good approximation of (1) by passing such an object to the constructor

```
TwoDimensionalFunctionIntegration(UniformRandomNumberSequence sequenceGenerator,
                                  DoubleUnaryOperator integrand).
```

(b) In order to solve this issue, we can add a method

```
double[][] getTwoDimensionalSequenceOfUncorrelatedRandomNumbers(),
```

to the interface `UniformRandomNumberSequence`, which returns a matrix $A = (a_{i,j})_{i=1,2,j=1,\dots,N}$ with the requirement that $a_{1,j}$ and $a_{2,j}$ are uncorrelated for any $j = 1, \dots, n$. Any idea about a possible way to implement such a method?