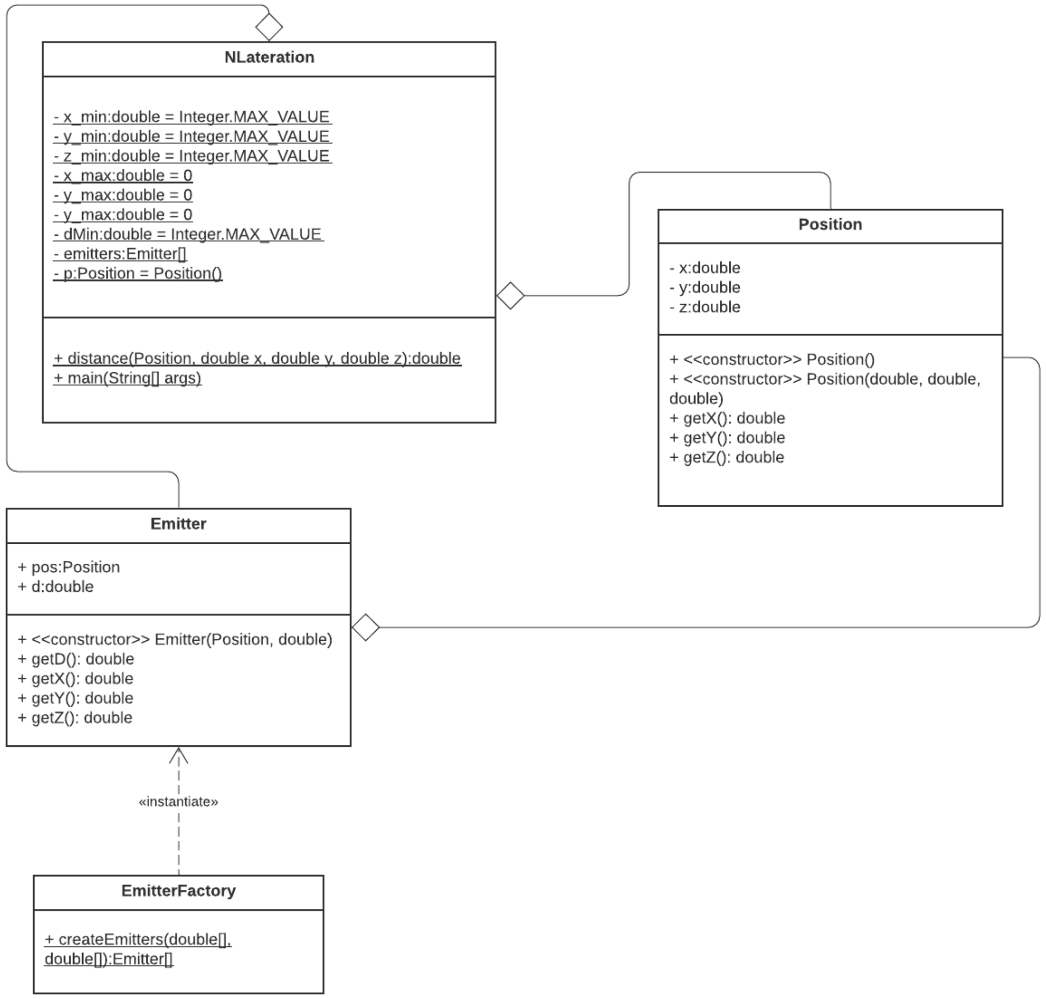
NLateration

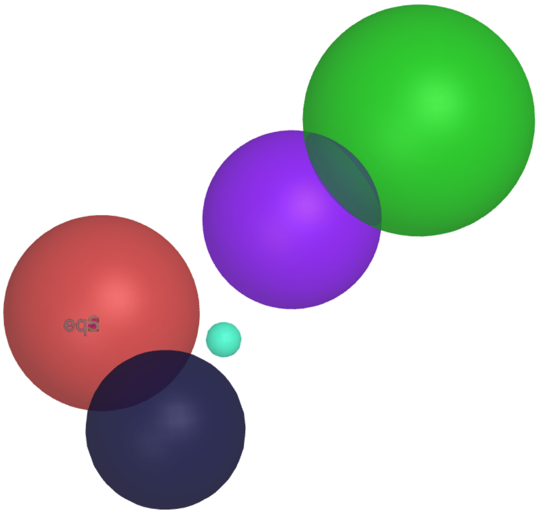
UML Diagram


NLateration
- x_min:double = Integer.MAX_VALUE
- y_min:double = Integer.MAX_VALUE
- z_min:double = Integer.MAX_VALUE
- x_max:double = 0
- y_max:double = 0
- y_max:double = 0
- dMin:double = Integer.MAX_VALUE
- emitters:Emitter[]
- p:Position = Position()
+ distance(Position, double x, double y, double z):double
+ main(String[] args)
Position
- x:double
- y:double
- z:double
+ <<constructor>> Position()
+ <<constructor>> Position(double, double, double)
+ getX(): double
+ getY(): double
+ getZ(): double
Emitter
+ pos:Position
+ d:double
+ <<constructor>> Emitter(Position, double)
+ getD(): double
+ getX(): double
+ getY(): double
+ getZ(): double
«instantiate»
EmitterFactory
+ createEmitters(double[], double[]):Emitter[]

4 Spheres:

step	X	Y	Z	time(ms)
0.5	3.5	1.0	1.5	4
0.1	3.3	1.5	1.1	43
0.05	3.25	0.95	1.6	184
0.01	3.26	0.96	1.59	17501

Geometrical view:



mint small sphere -> the result point (drawn as a sphere to be easily shown on the image, step = 0.1)

red sphere -> the first given sphere : $(x - 0.5)^2 + (y - 0.5)^2 + (z - 0.5)^2 = 3$

black sphere -> the second given sphere : $(x - 4)^2 + y^2 + z^2 = 2$

green sphere -> the third given sphere: $(x - 4)^2 + (y - 5)^2 + (z - 5)^2 = 4.2$

purple sphere -> the last given sphere: $(x - 3)^2 + (y - 3)^2 + (z - 3)^2 = 2.5$

3 Spheres(last sphere removed):

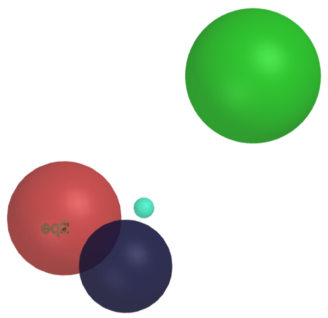
step	X	Y	Z	time(ms)
0.5	3.5	1.5	1.5	5
0.1	3.3	1.3	1.4	29
0.05	3.25	1.35	1.35	151
0.01	3.27	1.32	1.32	13839

mint small sphere -> the result point (drawn as a sphere to be easily shown on the image, step = 0.1)

red sphere -> the first given sphere : $(x - 0.5)^2 + (y - 0.5)^2 + (z - 0.5)^2 = 3$

black sphere -> the second given sphere : $(x - 4)^2 + y^2 + z^2 = 2$

green sphere -> the third given sphere: $(x - 4)^2 + (y - 5)^2 + (z - 5)^2 = 4.2$



3 Spheres(third sphere removed):

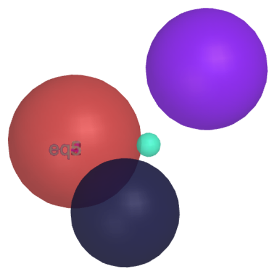
step	X	Y	Z	time(ms)
0.5	3.0	1.0	1.5	2
0.1	3.3	1.1	1.4	22
0.05	3.25	0.95	1.6	151
0.01	3.26	0.95	1.59	5220

mint small sphere -> the result point (drawn as a sphere to be easily shown on the image, step = 0.1)

red sphere -> the first given sphere : $(x - 0.5)^2 + (y - 0.5)^2 + (z - 0.5)^2 = 3$

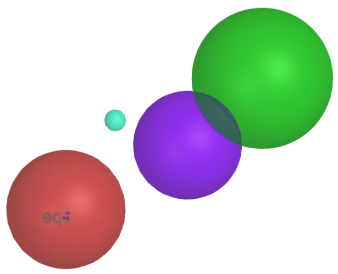
black sphere -> the second given sphere : $(x - 4)^2 + y^2 + z^2 = 2$

purple sphere -> the last given sphere: $(x - 3)^2 + (y - 3)^2 + (z - 3)^2 = 2.5$



3 Spheres(second sphere removed):

step	X	Y	Z	time(ms)
0.5	0.5	2.5	3.0	4
0.1	0.6	2.3	2.9	29
0.05	0.6	2.3	2.9	125
0.01	0.56	2.56	2.68	13829



mint small sphere -> the result point (drawn as a sphere to be easily shown on the image, step = 0.1)

red sphere -> the first given sphere : $(x - 0.5)^2 + (y - 0.5)^2 + (z - 0.5)^2 = 3$

black sphere -> the second given sphere : $(x - 4)^2 + y^2 + z^2 = 2$

green sphere -> the third given sphere: $(x - 4)^2 + (y - 5)^2 + (z - 5)^2 = 4.2$

3 Spheres(first sphere removed):

step	X	Y	Z	time(ms)
0.5	4.3	1.5	1.5	4
0.1	4.2	1.4	1.5	28

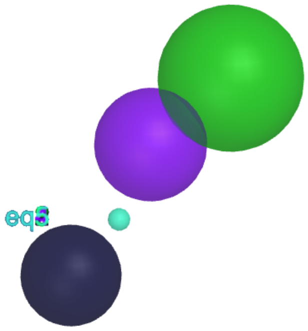
0.05	4.2	1.45	1.45	99
0.01	4.11	1.41	1.42	10133

mint small sphere -> the result point (drawn as a sphere to be easily shown on the image, step = 0.1)

black sphere -> the second given sphere : $(x - 4)^2 + y^2 + z^2 = 2$

green sphere -> the third given sphere: $(x - 4)^2 + (y - 5)^2 + (z - 5)^2 = 4.2$

purple sphere -> the last given sphere: $(x - 3)^2 + (y - 3)^2 + (z - 3)^2 = 2.5$



Based on results that we got by changing the value of step, we can claim that slight differences of that value don’t have huge impact on the point.

Specs

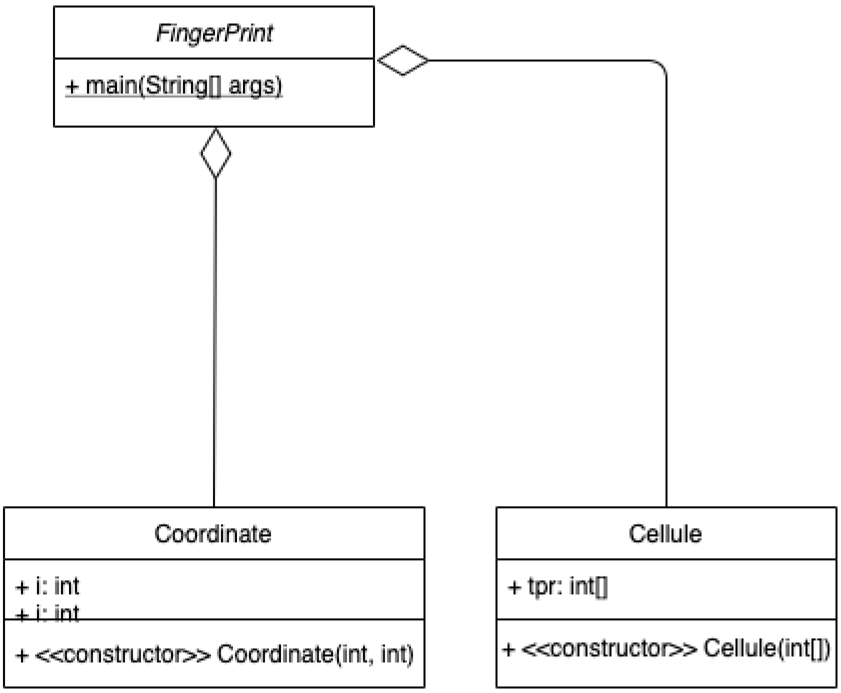
We used specific mathematical tool called Geogebra for demonstration purposes.
The problem was implemented in IntelliJ IDEA 2020.3.2 (Ultimate Edition) using Java programing language on MacOS operating system.

Data Specification

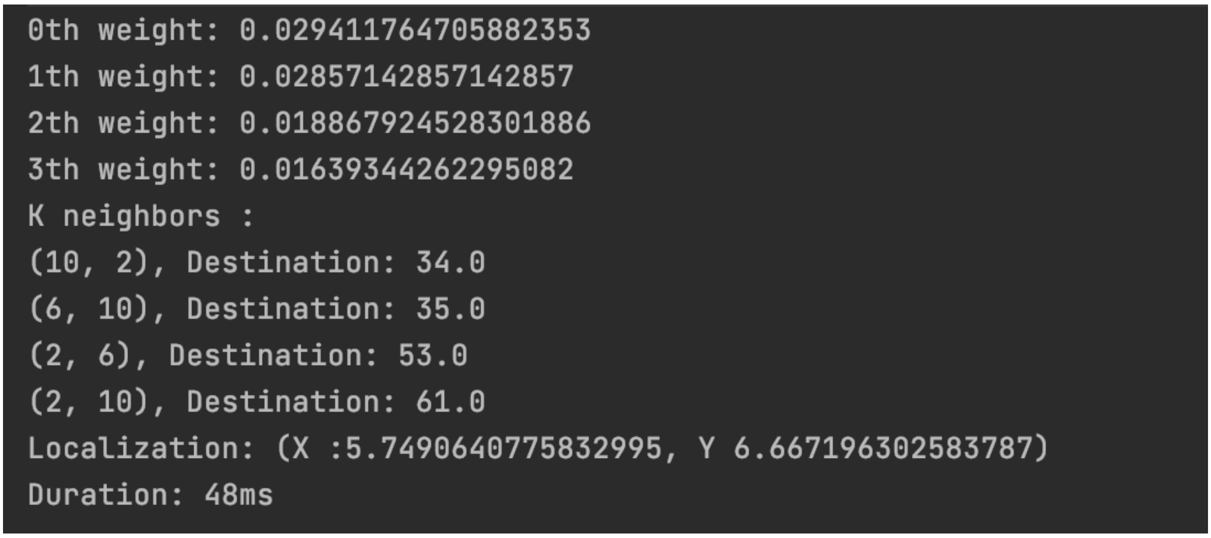
As you can see in the implementation, we changed the logic of getting the emetteurs by creating a factory class that takes arrays of positions and distances. That makes the project more flexible, cause we may change emetteurs initialization in future.
Additionally, while running the project the user can see the amount of time spent on running.

FingerPrint

UML Diagram



Output we got:



```
0th weight: 0.029411764705882353
1th weight: 0.02857142857142857
2th weight: 0.018867924528301886
3th weight: 0.01639344262295082
K neighbors :
(10, 2), Destination: 34.0
(6, 10), Destination: 35.0
(2, 6), Destination: 53.0
(2, 10), Destination: 61.0
Localization: (X :5.7490640775832995, Y 6.667196302583787)
Duration: 48ms
```

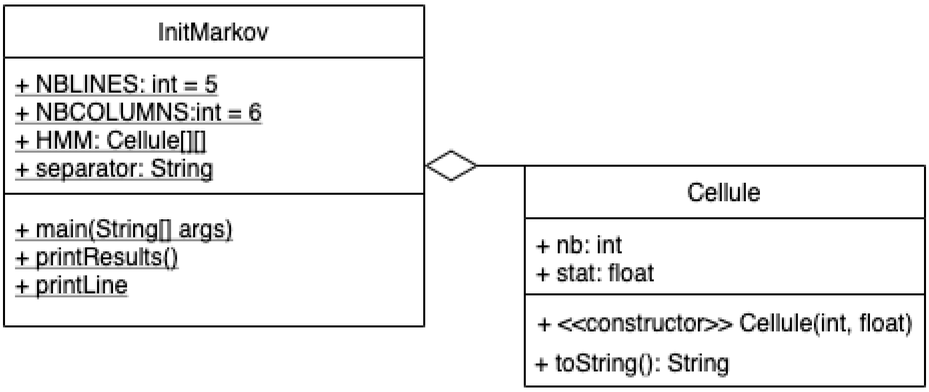
Specs:

The problem was implemented in IntelliJ IDEA 2020.3.2 (Ultimate Edition) using Java programing language on MacOS operating system.

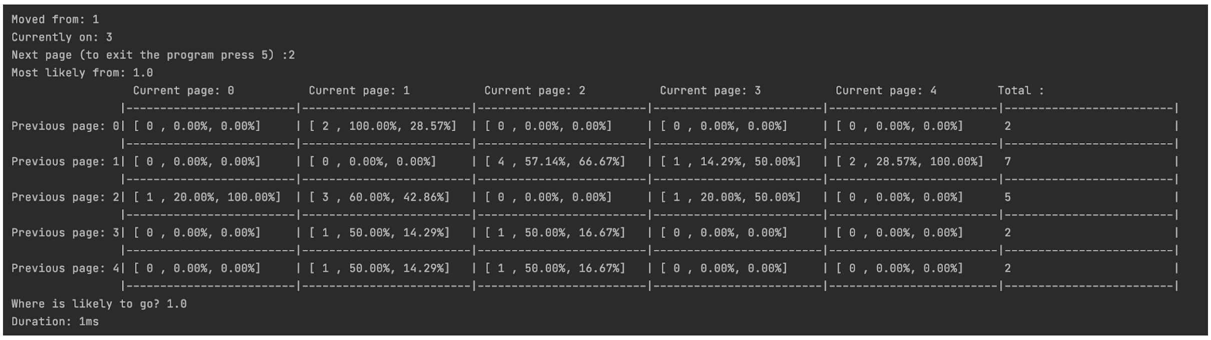
Data spec: used hashmap, arraylist and custom comparator based on euclidean destination

Markov

UML Diagram



Output we got:



```
Moved from: 1
Currently on: 3
Next page (to exit the program press 5) :2
Most likely from: 1.0
                  Current page: 0          Current page: 1          Current page: 2          Current page: 3          Current page: 4        Total :
                |-------------------------|-------------------------|-------------------------|-------------------------|-------------------------|-------------------------|
Previous page: 0| [ 0 , 0.00%, 0.00%]     | [ 2 , 100.00%, 28.57%]  | [ 0 , 0.00%, 0.00%]     | [ 0 , 0.00%, 0.00%]     | [ 0 , 0.00%, 0.00%]      2                        |
                |-------------------------|-------------------------|-------------------------|-------------------------|-------------------------|-------------------------|
Previous page: 1| [ 0 , 0.00%, 0.00%]     | [ 0 , 0.00%, 0.00%]     | [ 4 , 57.14%, 66.67%]   | [ 1 , 14.29%, 50.00%]   | [ 2 , 28.57%, 100.00%]   7                        |
                |-------------------------|-------------------------|-------------------------|-------------------------|-------------------------|-------------------------|
Previous page: 2| [ 1 , 20.00%, 100.00%]  | [ 3 , 60.00%, 42.86%]   | [ 0 , 0.00%, 0.00%]     | [ 1 , 20.00%, 50.00%]   | [ 0 , 0.00%, 0.00%]      5                        |
                |-------------------------|-------------------------|-------------------------|-------------------------|-------------------------|-------------------------|
Previous page: 3| [ 0 , 0.00%, 0.00%]     | [ 1 , 50.00%, 14.29%]   | [ 1 , 50.00%, 16.67%]   | [ 0 , 0.00%, 0.00%]     | [ 0 , 0.00%, 0.00%]      2                        |
                |-------------------------|-------------------------|-------------------------|-------------------------|-------------------------|-------------------------|
Previous page: 4| [ 0 , 0.00%, 0.00%]     | [ 1 , 50.00%, 14.29%]   | [ 1 , 50.00%, 16.67%]   | [ 0 , 0.00%, 0.00%]     | [ 0 , 0.00%, 0.00%]      2                        |
                |-------------------------|-------------------------|-------------------------|-------------------------|-------------------------|-------------------------|
Where is likely to go? 1.0
Duration: 1ms
```

Here you can see that were additionally calculated most probable next position and previous position based on current position.

Specs:

The problem was implemented in IntelliJ IDEA 2020.3.2 (Ultimate Edition) using Java programing language on MacOS operating system.

Average duration $\approx 5ms$