

REPORT PRÀCTICA 1

1. ***An introduction where the problem is described. For example, what should the program do? Which classes do you have to define? Which methods do you have to define? Which methods do you have to implement for these classes?***

L'objectiu d'aquesta primera pràctica ha estat la creació de diverses classes, amb funcions concretes per representar punts geomètrics i matrius. Com a visió general de la pràctica, s'ha encarat com un exercici d'anàlisi de distàncies entre ciutats. És a dir, les instàncies de classe GeometricPoint estan directament relacionades amb el concepte de ciutats com unitats, mentre que les instàncies de classe Matrix es traslladen conceptualment com taules que relacionen distàncies entre ciutats. Tot i així i amb ajuda del treball realitzat al seminari 1, hem hagut de definir i programar tota una sèrie de classes (amb els seus atributs i mètodes), funcions, implementacions i relacions entre aquestes per fer funcionar el nostre programa. A continuació desenvoluparem aquestes explicacions:

1. CREACIÓ DE GEOMETRICPOINT

Aquest ha estat el primer mòdul a definir. Tal com vam fer en el seminari (de forma conceptual), hem creat una classe de tipus GeometricPoint amb unes coordenades X i Y i un nom com atributs. Com sabem de teoria, els atributs els hem declarat privats. Seguint les funcions, hem definit els mètodes. Primerament tenim el constructor, seguit dels getters (per retornar el valor actual de l'atribut X, Y i nom) i els setters (per canviar el valor dels atributs esmentats anteriorment). Un cop definits els mètodes especials, hem creat dos mètodes addicionals, necessaris pel correcte funcionament de la pràctica.

- a. distancePoint: En aquest mètode ens convé poder calcular la distància **pitagòrica** entre dos punts. Com estem definint els mètodes de la classe GeometricPoint, només necessitem passar per paràmetre un punt (amb els seus atributs), ja que les dades de l'altre punt les agafem dels valors actuals dels atributs de la classe. Aquest mètode és important ja que posteriorment, en implementar-ho en altres mètodes, en podrem obtenir la distància entre dos ciutats, element imprescindible per la matriu de distàncies que volem definir.
- b. printPoint: Aquesta mètode, amb la funció ja implementada en Visual Studio Code de print, ens imprimeix per pantalla qualsevol variable de classe GeometricPoint.

2. CREACIÓ I IMPLEMENTACIÓ DE DISTANCEMATRIX

En aquest segon mòdul, definim la classe DistanceMatrix, amb els seus atributs i mètodes corresponents. Veiem però que en aquesta classe no només hem hagut de definir atributs i mètodes propis de DistanceMatrix, sinó que també hem implementat mètodes que pertanyen a la classe GeometricPoint.

Veiem que els atributs es tracten d'una LinkedList de tipus GeometricPoint (cities) i d'una LinkedList de LinkedLists de tipus Double (matrix). Hem utilitzat double en comptes de float

ja que, tot i que volem guardar nombres decimals, double utilitza 64 bits per guardar una variable, pel que pot guardar nombres més grans.

D'altra banda, els mètodes especials (constructor, getters i setters) també apareixen. Tenen la mateixa funció que els que hem definit a GeometricPoint però ho fan amb DistanceMatrix. Als getters, hem definit tres mètodes que retornen diferents aspectes de la matriu.

- a. **getCityName** retorna el nom de la ciutat, obtinguda amb un índex passat per paràmetre. En aquest mètode implementem getName, definit a GeometricPoint.
- b. **getNoOfCities** retorna el nombre de ciutats definides / afegides (vindria a ser el nombre de variables de classe GeometricPoint creades). En aquest mètode utilitzem la funció size(), predefinida a la nostra IDE.
- c. **getDistance** retorna la distància entre dues ciutats. Passem per paràmetre també dos índexs, per obtenir les ciutats. En aquest mètode implementem distancePoint, definit a GeometricPoint.

Com a mètodes addicionals, hem definit:

- d. addCity: Afegim a la llista cities (atribut de DistanceMatrix) una nova ciutat, la qual podrem posteriorment incloure en la nostra matriu de distàncies (que vindria a ser part del nostre objectiu principal).
- e. createDistanceMatrix: Aquest mètode és clau pel correcte funcionament de la pràctica. Primer creem una matriu de tipus LinkedList de LinkedLists de tipus Double. Això ho fem principalment per no manipular la matriu original. Implementem la funció getNoOfCities i guardem el resultat en una variable que és la que ens marca el final de bucle per iterar les files i columnes de la matriu. Amb els dos bucles (ja que la matriu és de 2 dimensions), iterem totes les posicions afegint el valor obtingut a la crida de la funció getDistance. Un cop hem iterat tota la matriu i els valors estan afegits, traslladem la matriu a la nostra variable original (l'atribut matrix).

3. MÒDULS DE TESTEIG

Com a última part de la nostra pràctica, hem creat 3 mòduls de testeig per tal de provar el nostre programa i assegurar-nos que funciona de forma correcta.

- a. TestPoint: En aquest mòdul definim quatre instàncies (ciutats) diferents de classe GeometricPoint i fem diferent testejos per comprovar els mètodes. Implementem tots els mètodes de la classe: GeometricPoint, getX, getY, getName, setX, setY, setName, printPoint i distancePoint.
- b. TestDistanceMatrix: El mateix fem en aquest mòdul. Tot i així, a l'hora de crear les instàncies, les definim directament en el mètode addCity. Provem els getters i setters i les funcions addCity i createDistanceMatrix.
- c. TestDisplayMatrix: Aquest mòdul ja venia inclòs en el pdf proporcionat. Forma part de la part opcional i té com a objectiu crear de forma interactiva la matriu de

distàncies. Per relacionar el DisplayMatrix amb el nostre codi, hem inclòs tots els mòduls en un package comú, anomenat **distancematrix**.

2. A description of possible alternative solutions that were discussed, and a description of the chosen solution and the reason for choosing this solution rather than others. It is also a good idea to mention the related theoretical concepts of object-oriented programming that were applied as part of the solution.

Durant la pràctica, ens hem trobat amb diverses dificultats i incògnites que ens han fet redefinir certs mètodes. Principalment, no vam utilitzar en cap moment la funció this, però, com la vam veure a teoria, hem decidit implementar-la en certs mètodes per optimitzar el codi.

D'altra banda, vam tenir certs problemes amb certs mètodes com printPoint i addCity:

- A printPoint, inicialment, volíem passar un punt de tipus GeometricPoint com a paràmetre. Vam veure que a l'hora d'utilitzar la funció havíem d'especificar el paràmetre cada cop que volíem printar una ciutat. També teníem el mateix problema si passàvem els atributs de GeometricPoint com paràmetres de la funció. Per tant, el que vam fer per optimitzar el programa, va ser no afegir cap paràmetre a printPoint i, a l'hora de printejar, fer-ho amb el nom de la ciutat ja creada (nomdelaciutat.printPoint()).
- A addCity ens va passar el contrari. Principalment, volíem fer-ho de la mateixa manera que amb printPoint. Tot i així, en veure que al pdf especificava que, si volíem que el DisplayMatrix funcionés correctament havíem de realitzar els diferents mètodes tal i com posava en el fitxer, vam decidir incloure els atributs de GeometricPoint com paràmetres, simplement, per comoditat respecte el codi de DisplayMatrix. Addicionalment, vam provar de passar per paràmetre un punt de tipus GeometricPoint, però això implicava errors a DisplayMatrix que no sabíem solucionar.

Altrament, els atributs de DistanceMatrix els vam declarar inicialment com arrays i no llistes, però vam veure que era més fàcil manipular llistes i els vam acabar canviant.

Relacionat amb els conceptes teòrics, podem veure que entre les classes GeometricPoint i DistanceMatrix hi ha una relació d'ús o dependència (unidireccional). La classe independent seria GeometricPoint i la dependent DistanceMatrix, ja que aquesta utilitza instàncies de tipus GeometricPoint.

3. A conclusion that describes how well the solution worked in practice, i.e. did the tests show that the classes were correctly implemented? You can also mention any difficulties during the implementation as well as any doubts you might have had.

Volem destacar que hem tingut certs problemes a l'hora d'obrir el projecte, no ens funcionava de primeres. Ens sortia un error de package. Finalment, ho hem arreglat creant

una carpeta directament al Visual Studio Code amb el nom de distancematrix i incloent tots els mòduls dins la carpeta.

Concloent amb la pràctica, gràcies als mòduls de testeig, hem pogut comprovar que la pràctica ens funciona correctament. Cap test ens dona error (excepte l'error de package el qual hem parlat anteriorment com solucionar-ho) i, a TestDisplayMatrix, quan carrega la matriu, accepta només nombres racionals (positius i negatius) com a coordenades. Per tant, considerem que no hem trobat cap error en la nostra pràctica.

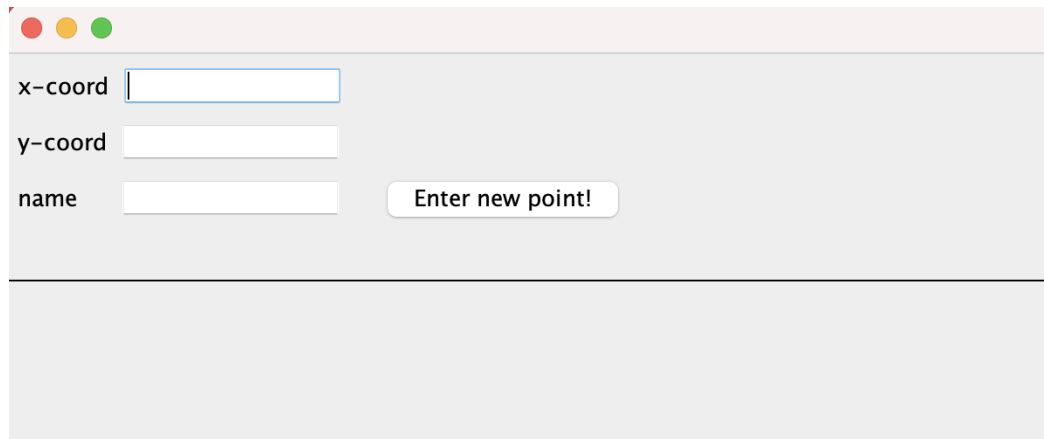
TESTPOINT:

```
Les coordenades de les diferents ciutats són:  
Barcelona = (3.0,7.0)  
Pontevedra = (5.0,2.0)  
Madrid = (0.0,0.0)  
València = (4.0,6.0)  
La distància entre Barcelona i València és de: 5.385164807134504 km.  
La distància entre Madrid i València és de: 7.211102550927978 km.  
El nou nom de la ciutat de Barcelona és: Tarragona  
Les seves noves coordenades són:  
Tarragona = (6.0,10.0)
```

DISTANCEMATRIX:

```
En la matriu hi ha 4 ciutats.  
La distància entre Madrid i Pontevedra és de: 5.385164807134504 km.  
La matriu creada és:  
[[0.0, 5.385164807134504, 7.615773105863909, 1.4142135623730951], [5.385164807134504, 0.0, 5.385164807134504, 4.123105625617661], [7.615773105863909, 5.385164807134504, 0.0, 7.211102550927978], [1.4142135623730951, 4.123105625617661, 7.211102550927978, 0.0]]
```

DISPLAYMATRIX:



Afegim coordenades de les ciutats:

- Barcelona = (2, 1)
- Madrid = (3, 4)
- Valencia = (5, 3)
- Pontevedra = (8'5,7'2)

	Barcelona	Madrid	València	Pontevedra
Barcelona	0,00	3,16	3,61	8,98
Madrid	3,16	0,00	2,24	6,36
València	3,61	2,24	0,00	5,47
Pontevedra	8,98	6,36	5,47	0,00

Veiem l'error a l'hora de ficar una lletra o un caràcter especial:

x-coord
y-coord
name

Wrong coordinate format