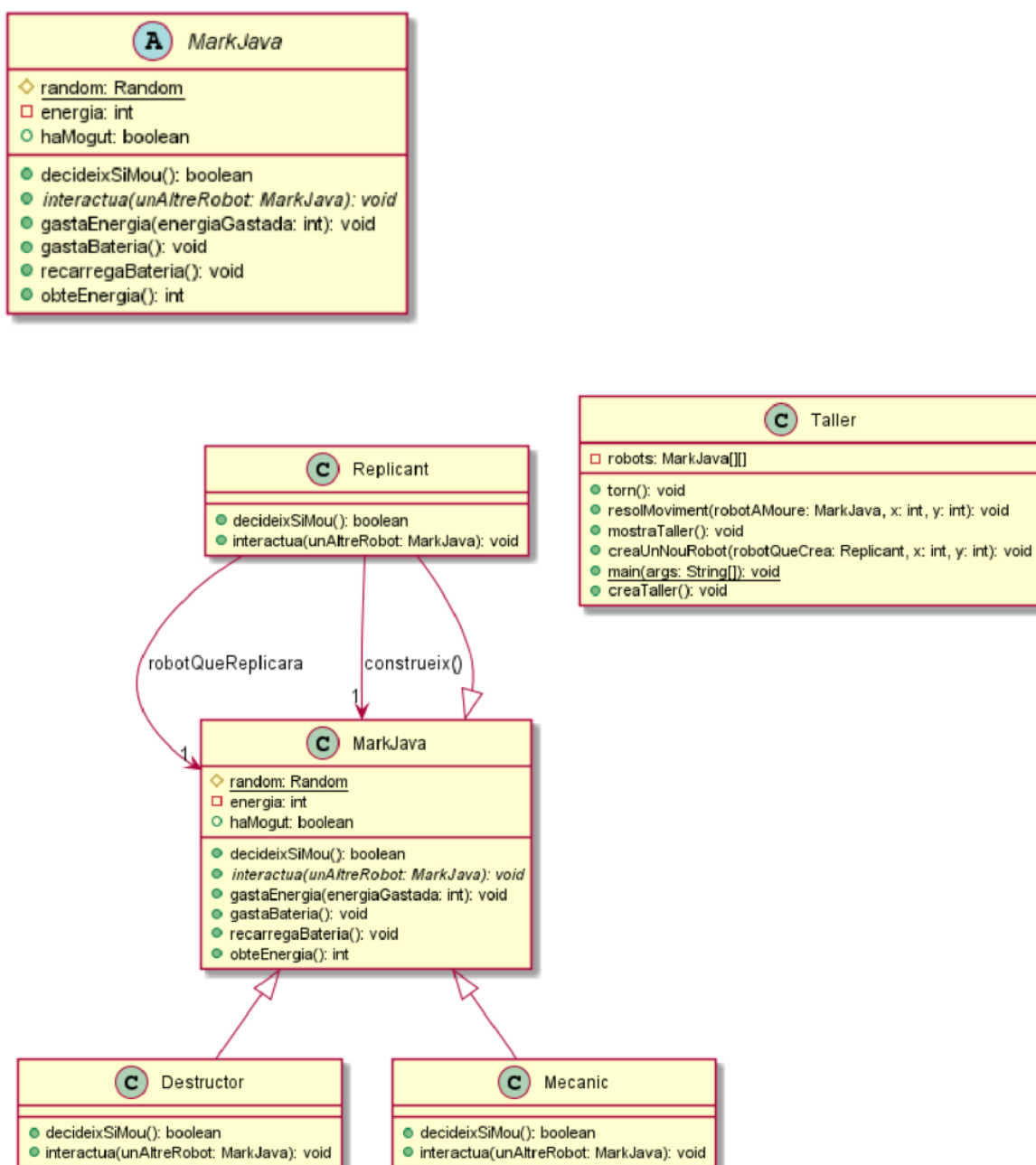

 Institut Sabadell	CFGS DAM - DAM vi M03-UF4. Programació Orientada a Objectes + Intro M05-UF3. Disseny OO	Curs 2020 / 2021
--	--	------------------

* Aquest exercici pertany als materials del professor Joan Queralt (M03-UF4)

Crea un projecte “robots” que implementi el següent diagrama de classes i les especificacions posteriors.



 Institut Sabadell	CFGS DAM - DAM vi M03-UF4. Programació Orientada a Objectes + Intro M05-UF3. Disseny OO	Curs 2020 / 2021
--	--	------------------

Els robots


Els robots de la classe *MarkJava* van ser dels més famosos de la història. N'hi havia de tres tipus: els *Destructors*, els *Replicants* i els *Mecànics*. Tots els robots *MarkJava* estaven alimentats per una bateria que els donava energia. Els robots només podien fer les seves accions si tenien prou energia. Els *Destructors* buidaven les bateries dels seus enemics, deixant-los immòbils, els *Replicants* eren capaços de construir altres robots i els *Mecànics* eren capaços de recarregar la seva pròpia bateria i la dels altres robots.

Classe *MarkJava*

Dissenya la classe *MarkJava* amb les següents propietats:

Atributs

- *Random random*: s'utilitzarà per generar nombres aleatoris quan calgui.
- *int energia*: l'energia que té el robot per seguir actuant. Tots els robots comencen amb una energia de 10.
- *boolean haMogut*: indicarà si ja ha mogut o no durant un torn.

 Institut Sabadell	CFGS DAM - DAM vi M03-UF4. Programació Orientada a Objectes + Intro M05-UF3. Disseny OO	Curs 2020 / 2021
--	--	------------------

Mètodes


- *int obteEnergia()*: retorna l'atribut *energia*.
- *boolean decideixSiMou()*: Tots els robots, retornaran directament *false* si ja han mogut aquest torn (*haMogut* és *true*), i posaran a *true* l'atribut *haMogut* cada cop que es cridi aquest mètode. Torna *true* si el robot es vol moure i *false* si no, però cada tipus ho acaba de decidir (a més de pel torn) de forma diferent.
- *void gastaEnergia(int energiaGastada)* aquest mètode resta l'*energiaGastada* de l'*energia* del robot. Si l'*energia* queda menor que 0 es posarà a 0. Sempre que un robot gastí energia ho farà cridant a aquest mètode.
- *void gastaBateria()*: aquest mètode deixa l'energia a 0.
- *void interactua(MarkJava unAltreRobot)*: aquest mètode defineix què passa quan aquest robot en troba un altre, però cada tipus fa coses diferents.
- *void recarregaBateria()*: torna la bateria al seu valor màxim.

Classe *Destructor*


Dissenya la classe *Destructor* amb les següents propietats:

Mètodes

- *void interactua(MarkJava unAltreRobot)*: si el robot actual (sobre el qual s'executa el mètode) té menys de 5 d'energia no fa res i es retorna del mètode. En cas contrari, el robot *Destructor* ataca als altres robots quan els troba. Si l'altre robot no és *Destructor* li gasta la bateria (de l'altre), l'atac té èxit. Si el robot que ha trobat és de tipus *Destructor*, hi ha un 50% de possibilitats que l'atac tingui èxit (o sigui, s'ha de sortejar entre dos valors possibles si tindrà èxit o no). Si l'atac té èxit li gasta la bateria al robot atacat. En qualsevol cas, si el robot ataca, perd 3 punts d'energia. Tenir èxit vol dir que li passa alguna cosa al que rep l'atac. Si l'atac fracassa aquest mètode no fa res.
- *boolean decideixSiMou()*: Encara que a un robot *Destructor* li toqui moure's per torn (allò ja està implementat a la superclasse) decidirà que no es mou si la seva energia és inferior a 5. En cas contrari, es treu un nombre aleatori entre 1 i 10. Si el resultat

 Institut Sabadell	CFGS DAM - DAM vi M03-UF4. Programació Orientada a Objectes + Intro M05-UF3. Disseny OO	Curs 2020 / 2021
--	--	------------------

és menor o igual que 4 es retorna *true* (decideix que sí es mou), en cas contrari es retorna *false*. Si no li tocava moure's per torn també retornarà *false*.

 Institut Sabadell	CFGS DAM - DAM vi M03-UF4. Programació Orientada a Objectes + Intro M05-UF3. Disseny OO	Curs 2020 / 2021
--	--	------------------

Classe *Mecànic*

Dissenya la classe *Mecanic* amb les següents propietats:

Mètodes

- `void interactua(MarkJava unAltreRobot)`: si el robot actual té una energia superior a 4, crida el mètode *recarregaBateria* de l'altre robot i ell perd un punt d'energia. En cas contrari, no fa res.
- `boolean decideixSiMou()`: Malgrat li toqui moure's per torn (implementat a la superclasse) si l'energia és menor de 5, el robot decideix no es mou, però crida el seu mètode *recarregaBateria*. Si li toca moure's i la seva energia és major o igual a 5, es treu un nombre aleatori entre 1 i 10. Si el resultat és menor o igual que 6 es retorna *true* (decideix que es mou). En qualsevol altre cas es retorna *false* (no es mou).

Classe *Replicant*


Dissenya la classe *Replicant* amb les següents propietats:

Atributs


- *MarkJava robotQueReplicara*: aquest atribut guardarà l'últim robot amb qui s'hagi trobat, per saber de quin tipus serà el robot que faci quan en creï un.

Mètodes

- `void interactua(MarkJava unAltreRobot)`: assigna el robot trobat a *robotQueReplicara*.
- `boolean decideixSiMou()`: Si no li toca moure's per torn, retorna *false*. Encara que sí el toqui moure's, si l'energia és menor de 7, el robot no es mou (torna false). Si no, es treu un nombre aleatori entre 1 i 10. Si el resultat és menor o igual que 6 es retorna *true*. en cas contrari es retorna *false*.
- `MarkJava construeix()`: Si l'energia del robot actual és menor de 7 retorna *null*. Si no, crea un nou robot del mateix tipus que el que tingui guardat a *robotQueReplicara* i el

	<p>CFGS DAM - DAM vi</p> <p>M03-UF4. Programació Orientada a Objectes + Intro M05-UF3. Disseny OO</p>	<p>Curs 2020 / 2021</p>
---	---	--------------------------------

retorna. Si *robotQueReplicara* val *null*, crea un robot de tipus aleatori (tots tenen la mateixa probabilitat). Si ha construït algun robot, ell perd 5 d'energia.

 Institut Sabadell	CFGS DAM - DAM vi M03-UF4. Programació Orientada a Objectes + Intro M05-UF3. Disseny OO	Curs 2020 / 2021
--	--	------------------

Classe *Taller*

La classe *Taller* té les següents propietats:


Atributs

- *MarkJava[][] robots*: aquesta matriu representa el taller (que és on viuen els robots), i té unes dimensions de 5x5. A cada posició hi podrà haver un robot o no, valdrà *null* si no hi ha cap robot en aquella posició.

Mètodes

- *public static void main(String[] args)*: el mètode *main* crea un objecte de tipus *Taller* i crida el seu mètode *creaTaller*.
- *creaTaller()*: crea a la classe *Taller* un mètode *creaTaller* que posi alguns robots en ell (la posició, tipus i quantitat podeu posar la que vulgueu, però almenys un robot de cada tipus), que mostri l'estat inicial del taller i que cridi el mètode *torn* d'aquest taller 10 vegades.
- *void torn()*: recorre una vegada la matriu robots, per cada robot es crida el seu mètode *decideixSiMou* i es mira el resultat. Si és *true* es crida el mètode *resolMoviment* amb el robot que s'ha de moure i la seva posició com a paràmetres. Quan s'ha recorregut tota la matriu, es torna a recórrer, posant l'atribut *haMogut* de tots els robots a *false*. Finalment, es crida el mètode *mostraTaller*.
- *void resolMoviment(MarkJava robotAMoure, int x, int y)*: Els valors *x* i *y* són les coordenades de la cel·la on es troba ara el robot. Es treu un nombre aleatori del 0 al 3 per saber en quina direcció es vol moure el robot. El 0 vol dir dreta, l'1 vol dir avall, el 2 esquerra i el 3 amunt. Si a causa del seu moviment el robot queda fora del taller, el robot no es mou. En cas contrari, es resta 1 d'energia al robot i es mira què hi ha a la casella on vol anar.

Si no hi ha res, es canvia la posició del robot a la nova casella. Si el robot és de tipus Replicant crearà (*creaUnNouRobot*) una rèplica de sí mateix i la deixarà a la posició que ocupava ell abans de moure's.

 Institut Sabadell	CFGS DAM - DAM vi M03-UF4. Programació Orientada a Objectes + Intro M05-UF3. Disseny OO	Curs 2020 / 2021
--	--	------------------

Si hi ha un altre robot, es crida al mètode interactuar del *robotAMoure* passant com a paràmetre el robot que s'ha trobat.

En qualsevol cas, si un robot topa amb un altre, no es mou.

- *void creaUnNouRobot(Replicant robotQueCrea, int x, int y)*: s'assigna a *robots[x][y]* el resultat de cridar el mètode *construeix* de *robotQueCrea*.
- *void mostraTaller()*: mostra l'estat del taller per pantalla. Això significa que es dibuixarà un quadrat amb lletres per pantalla on cada lletra representa una posició de la matriu robots. Un - indica que la posició és buida, una **D** indica un destructor, una **R** un replicant i una **M** un mecànic.

Consells

- Pensa que no tens enlloc el “nom del tipus del robot”, amb el que l'única forma de poder distingir els tres tipus és mitjançant la seva classe (instanceof)
- Qualsevol crida a una funció que torni null ha de tenir un if per comprovar si el valor és null. Altrament en accedir al valor tirarà una *NullPointerException*