PROVES IRIA

CLASSE AFECTACIÓVIRUSREGIÓ, VIRUS (ARN I RNA), REGIÓ I FAMILIAREGIÓ

Prova 1: Provo la funció avançarUnDia (ProvaIriaAfectacio):

Dia D-1
• totalMalalts = 1515
• totalMorts = 150
• totalInfectats = 3040

R.inf(V,D-1) INFECTATS = [100,90,80,90,100,111,110]
R.mal(V,D-1) MALALTS = [42, 45, 55, 50]
R.mor(V,D-1) MORTS = [a, b, c, d]
R.imm(V,D-1) IMMUNES= [97, 99, ...]

DIA D
Suposem que fent els càlculs indicats prèviament, ha resultat un total de 200 nous contagis.
*Nous_malalts(R,V,D) = R.inf(V,D-1)[V.I()] * V.probMalalt() = R.inf(V,D-1)[3] * V.probMalalt() = 80*0.5 = 40*
*Nous_morts(R,V,D) = Nous_malalts(R,V,D) * V.probMort() = 40 * 0.1 = 4*
Morts_per_dia(R,V,D) = Nous_morts(R,V,D) / V.duradaMalaltia() = 4 / 4 = 1
• totalMalalts = 1515 +40 = 1555
• totalMorts = 150 + 1 +a + b +c
• totalInfectats = 3040 + 200 = 3240

R.inf(V,D) INFECTATS = [200,100,90,80-1,90-a,100-b,111-c]
R.mal(V,D) MALALTS = [40-1, 42-a, 45-b, 55-c]
R.mor(V,D) MORTS = [1, a, b, c] (morts ja aplicats)
IMMUNES= [110, 97, 99, ...]

Per provar si la funció avançarUnDia() funciona bé, he decidit fer exactament l'exemple que surt a l'annex de fórmules. La idea era veure si, a partir dels valors inicials que ens dona l'exemple, la funció que he fet realment actualitza les dades com hauria de fer segons les fórmules.

Per fer-ho, m'he creat una classe de prova (la tinc penjada al GitHub) on primer he creat un virus amb exactament els mateixos valors que diu l'exemple (així també he aprofitat per comprovar que el constructor funciona bé). Després he hagut d'afegir uns quants mètodes per poder modificar les llistes que tinc internament a la classe AfectacioVirusRegio, i així posar-hi manualment els valors de infectats, malalts, morts… de l'exemple.

Un cop ho tenia tot preparat, he executat el mètode avançarUnDia() i he anat mirant si la sortida que em donava (tant les llistes com els totals) coincidia amb la que hauria de sortir segons els càlculs. En concret, he mirat si es feien bé els nous malalts i contagiosos, si es repartien correctament les morts entre dies, i si s'actualitzaven bé tots els comptadors.

- **Sortida de la classe de prova:**

```
== INFECTATS INICIALS ==
[90, 100, 111, 110]
== MALALTS INICIALS ==
[42, 45, 55, 50]
== TOTALS INICIALS ==
Total malalts: 1515
Total morts: 150
Total infectats: 3040
```

```
AVANÇEM UN DIA :)

== INFECTATS NO CONTAGIOSOS ==
[200, 100, 90]
== INFECTATS ==
[79, 90, 100, 111]
== MALALTS ==
[39, 42, 45, 55]
== MORTS PER DIA ==
[1, 0, 0, 0]
== TOTALS ==
Total malalts: 1555
Total morts: 151
Total infectats: 3240
```

La veritat que estic molt contenta ja que la sortida és exactament la sortida que s'esperava. Això em confirma que de moment, la classe funciona bé.

Si ara provem a avançar un altre dia, anem a veure la sortida:

```
AVANÇEM UN ALTRE DIA :)

== INFECTATS NO CONTAGIOSOS ==
[23, 200, 100]
== INFECTATS ==
[89, 78, 90, 100]
== MALALTS ==
[44, 38, 42, 45]
== MORTS PER DIA ==
[1, 1, 0, 0]
== TOTALS ==
Total malalts: 1600
Total morts: 153
Total infectats: 3263
```

Sembla que es segueix actualitzant correctament !!

Prova 2: Provo la funció de mutació per coincidència de la classe regió:

Suposem 2 virus A i B, de la mateixa família, i presents en una certa regió.
• Aquest dia hi ha hagut 3000 nous infectats d'A i 4000 de B. A, d'altra banda, és més fort que B.
• Suposem que el nombre d'infectats comuns és de 15, donant una PMC de 0.15.
Ara analitzem els 2 escenaris possibles (succeirà un o altre depenent de si el nombre aleatori generat és menor o major que 0.15):
a) Hi ha mutació
*• Apareix mutació per coincidència d'A i B, anomenada A_B_*
• Comprovem (per nom) si aquest virus ja existeix. Si existeix, no cal crear-lo de nou. Si no, el creem i l'afegim a la llista que es manté a simulació
*• Es crea una nova afectació per al virus A_B_ a la regió en qüestió. El nombre inicial d'infectats, seguint el full de fórmules, serà 15 * 0.15 (comuns * PMC) = 2 (arrodonit) (*)*
• Aquests 2 nous infectats s'han de restar d'A o de B... en realitat, es restaran de B, el més feble (a l'algorisme, de fet, diu això, però d'altra forma). Per tant, el nombre de nous contagis de B serà 3998 (tb cal modificar l'acumulat d'infectats d'aquesta afectació restant-hi 2).
(*) Si per casualitat ja existís afectació d'aquest virus en la regió, simplement li sumaríem els 2 nous infectats als nous infectats (i a l'acumulat).

b) No hi ha mutació
• Simplement, aquests 15 comuns d'A i de B "se'ls queda" el més fort, és a dir, A. Per tant, això equival a restar-los de B, que quedarà amb 3985 nous infectats (tb li restarem al seu acumulat d'infectats)

En aquesta segona prova volia veure si la mutació per coincidència funcionava correctament, segons el que diu l'enunciat i les fórmules de l'annex. Per fer-ho, he creat dos virus ARN diferents (el virus A i el virus B), i he fet que siguin de la mateixa família (FamiliaAuladell). Això m'ha anat molt bé per afirmar que si són de la mateixa família es dona la mutació, i si no ho son, no es fa.
He creat la regió IriaLand, i hi he afegit les dues afectacions amb els infectats que posava a l'exemple. Després, per tenir-ho tot controlat, he forçat els valors de cada afectació fent servir els mètodes de prova que havia creat abans.

Un cop fet això, bàsicament he cridat el mètode comprovarMutacionsPerCoincidencia() i he mirat per pantalla el resultat. Vull veure si realment s'ha afegit un nou virus mutat i si els infectats s'han distribuït correctament.

Amb aquesta prova he pogut veure si:

- només es produeix la mutació si hi ha coincidència de virus d'una mateixa família,
- es genera un nou virus amb nom correcte
- es reparteixen bé els infectats resultants de la mutació a la mateixa regió.

Ara bé, he tingut el problema de que clar, la mutació per coincidència es produeix segons un nombre aleatori. Per això, al començament, he format aquest nombre aleatori a 0, i així es produeix segur.

A més, no estic molt convençuda amb la fòrmula de → Inf_comuns(V,V',R,D) = (NCD(V,R,D)/R.poblacio(D)) * (NCD(V,R,D)/R.poblacio(D)) (*). Ja que sempre NCD serà menor a la població tota, i això donarà dos nombres menors que 1 fent que mai dongui més gran que 1 el resultat.

- **Sortida del codi provocant la mutació:**

```
=== RESULTATS DESPRES DE MUTACIO PER COINCIDENCIA ===
- Virus: A
  Infectats: [3000]
  Total infectats: 3000
  Total infectats: 3000
- Virus: B
  Infectats: [3998]
  Total infectats: 3998
  Total infectats: 3998
- Virus: A_B
  Infectats: [2]
  Total infectats: 2
  Total infectats: 2
```

La veritat que estic molt satisfeta perquè la sortida ha estat la que esperava

- **Sortida del codi sense provocar la mutació:**

```
=== RESULTATS DESPRÉS DE MUTACIÓ PER COINCIDÈNCIA ===
- Virus: A
  Infectats: [3000]
  Total infectats: 3000
  Total infectats: 3000
- Virus: B
  Infectats: [3985]
  Total infectats: 3985
  Total infectats: 3985
```

Prova 3: Provo els confinaments entre regions (ProvaConfinament):

En aquesta prova volia veure si els mètodes de confinaments que he fet funcionaven tal com m'havia imaginat, tant quan faig un confinament dur, com un confinament tou, i també quan el desfaig. El que volia comprovar és que realment es modifiquin les taxes de contacte, i que quan desconfinem, si el codi ha sigut capaç de guardar les originals i tot torni a estar com abans.

Per començar, he creat dues regions fictícies, Barcelona i Girona. Les he connectat amb una taxa de contacte externa de 0.3, per simular que hi ha moviment entre elles. Després he aplicat un confinament dur a Barcelona, que en teoria hauria de canviar la seva taxa interna i fer que no hi hagués cap mena de contacte amb les regions veïnes.

Un cop fet això, he pogut comprovar que per sort, la taxa interna havia baixat i que s'havia pogut tallar la connexió de Barcelona amb Girona (tant des de Barcelona com des de Girona). També he mirat que el booleà que indica si està confinada passava a true, i si, per sort tot ha funcionat !!.

Després, he cridat aixecarConfinament() per veure si realment es tornava a les taxes originals. I sí, el codi ha funcionat i les ha recuperat :). Això em confirma que els meus mètodes estan correctes i son capaços de guardar bé les dades d'abans del confinament i recuperar-les després.

També he volgut provar el confinament tou, que només evita el contacte entre dues regions. Aquesta vegada, he fet que Girona deixés de tenir contacte amb Barcelona. I ha anat perfecte, només s'ha canviat la taxa externa, i la resta no ha canviat.

Per últim, he fet servir el mètode hiHaConfinamentAmb() per comprovar si entre elles hi havia realment aïllament, i m'ha tornat true, que és exactament el que volia.

- **Sortida del codi:**

```
--- ABANS DE CONFINAR ---
Taxa interna BCN: 0.5
Taxa externa BCN->GIR: 0.3
Taxa externa GIR->BCN: 0.3

--- DESPRÉS DE CONFINAMENT DUR ---
Taxa interna BCN: 0.1
Taxa externa BCN->GIR: 0.0
Taxa externa GIR->BCN: 0.0
Barcelona confinada? true

--- DESPRÉS D'AIXECAR CONFINAMENT ---
Taxa interna BCN: 0.5
Taxa externa BCN->GIR: 0.3
Taxa externa GIR->BCN: 0.3
Barcelona confinada? false
```

```
--- DESPRÉS DE CONFINAMENT TOU GIR->BCN ---
Taxa externa BCN->GIR: 0.0
Taxa externa GIR->BCN: 0.0
Girona confinada? false

--- COMPROVACIÓ CONFINAMENT ENTRE REGIONS ---
Hi ha confinament entre BCN i GIR? true
```

Prova 4: Contagi entre regions amb i sense confinament (TestContagiAmbConfinament):

En aquesta prova volia comprovar si realment els confinaments tallaven el contagi entre dues regions, i que un cop es desconfina, aquest contagi torni a ser possible.

Per fer-ho, he creat dues regions connectades (RegioA i RegioB) i un virus fictici que ja havia fet servir en la prova 1. Per no haver de començar de zero, he reutilitzat exactament les mateixes dades d'infectats, contagiosos, malalts i morts que ja havia posat manualment abans, així les coneixia i era més fàcil analitzar els resultats de sortida. A més, he fet servir un mètode que m'he creat per afegir directament l'afectació amb aquests valors a RegioA.

Llavors, el que faig és:

- Avançar un dia amb les dues regions connectades i sense confinament, per veure si RegioB comença a rebre contagis.
- Després, aplico confinament dur a RegioA, que hauria d'evitar que es pugui contagiar cap regió veïna.
- Torno a avançar un dia i comprovo si RegioB s'ha mantingut igual.
- Finalment, desconfino RegioA i avanço un altre dia més, per veure si llavors sí que RegioB torna a rebre contagis.

- **Sortida del programa:**

```
--- SENSE CONFINAMENT ---
Contagiosos RegioA: 380
Infectats RegioB: 71
Poblacio RegioB: 10000

[Confinament aplicat a RegioA]

--- AMB CONFINAMENT ---
Contagiosos RegioA: 357
Infectats RegioB: 71
Poblacio RegioB: 10000

[Confinament aixecat a RegioA]

--- DESPRÉS DE DESCONFINAR ---
Contagiosos RegioA: 354
Infectats RegioB: 105
Poblacio RegioB: 10000
```

Un cop executada la prova, he anat mirant per pantalla l'evolució dels contagiosos a RegioA i els infectats a RegioB. Els resultats han estat exactament els que esperava, i estic super contenta perquè això em confirma que el confinament funciona com ha de funcionar.

Inicialment, sense cap tipus de confinament, ja es veu clarament que RegioB comença a rebre nous contagis (passa a tenir 71 infectats), degut a que RegioA té molts contagiosos. Després, quan aplico el confinament dur, el contagi extern es talla completament i RegioB es manté amb els mateixos infectats (71), cosa que confirma que no hi ha hagut cap transmissió nova.

Finalment, un cop aixeco el confinament i avanço un altre dia, RegioB torna a rebre contagis i puja fins als 105 infectats. Això em demostra que realment s'estava fent el confinament (barrera que evita contagis) i quan el desconfinem, restaurem les taxes i es torna a contagiar.

Per tant, m'ha sortit tot com esperava i he pogut validar que el funcionament dels confinaments l'he fet bé :).

Prova 5: Prova per verificar que el nom de les mutacions canvia correctament, i també els paràmetres

Tot i que és molt simple, aquesta prova l'he fet per assegurar-me que, quan un virus muta (ja sigui per error de còpia o per coincidència), el nom del nou virus es genera com toca i que es guarda a la llista de mutacions del virus original.

Per començar, he creat un virus base que he fet mutar diverses vegades per error de còpia, i així he pogut veure si sortien noms com VirusIria_1, VirusIria_2, VirusIria_3... tal com vull que passi. I ha funcionat perfecte!

Després també he fet una mutació per coincidència entre dos virus (VirusMireia i VirusRoman) per veure com es generava el nom combinat, i si realment quedava ben guardat dins la llista de mutacions dels dos. També m'ha servit per comprovar que cada cop que es repeteix la combinació, el nom es fa únic (_1, _2, etc.).

- **Sortida del codi:**

```
Mutacions per error de còpia:
- Mutacio 1: VirusIria_1
- Mutacio 2: VirusIria_2
- Mutacio 3: VirusIria_3
- Mutacio 4: VirusIria_4
- Mutacio 5: VirusIria_5

Mutacions per coincidència:
- Coincidència 1: VirusMireia_VirusRoman
- Coincidència 2: VirusMireia_VirusRoman_1
- Coincidència 3: VirusMireia_VirusRoman_2
```

```
Mutacions registrades a virusA:
- VirusMireia_VirusRoman
- VirusMireia_VirusRoman_1
- VirusMireia_VirusRoman_2

Mutacions registrades a virusOriginal:
- VirusIria_1
- VirusIria_2
- VirusIria_3
- VirusIria_4
- VirusIria_5
```

Tot ha funcionat perfecte, i veiem que per exemple al de mutació per coincidència, el nom es fa per ordre alfabètic i de manera adecuada.

- **Si ara veiem els valors dels virus mutats:**

Valors inicials:

```
new VirusARN( nom_virus: "VirusIria", fam, p_mal: 0.5, t_inc: 3, t_lat: 3, p_mor: 0.1, t_con: 4, p_con: 0.3, t_imm: 10, p_mutEC: 0.2);
```

```
VirusARN( nom_virus: "VirusMireia", fam, p_mal: 0.4, t_inc: 3, t_lat: 3, p_mor: 0.2, t_con: 5, p_con: 0.2, t_imm: 8, p_mutEC: 0.2);
VirusARN( nom_virus: "VirusRoman", fam, p_mal: 0.6, t_inc: 5, t_lat: 5, p_mor: 0.1, t_con: 3, p_con: 0.4, t_imm: 12, p_mutEC: 0.3);
```

Valors després de la mutació:

```
--- MUTACIÓ PER ERROR DE CÒPIA ---
Nom nou virus: VirusIria_1
pMal: 0.455
pMor: 0.084
tCon: 3
pCon: 0.342

--- MUTACIÓ PER COINCIDÈNCIA ---
Nom nou virus: VirusMireia_VirusRoman
pMal: 0.4480098807085835
pMor: 0.17015281954278483
tCon: 4
pCon: 0.3421894064860539
tInc: 4
tLat: 4
tImm: 10
pMutEC: 0.25
```

Podem veure com efectivament els de mutació per error de copia canvien en funció d'aquest 20%, afirmant així que està funcionant correctament. Després, els d'error per coincidència, també funciona com hauria de funcionar. Per una banda, els quatre primers paràmetres canvien en funció d'un nombre aleatori, mentres que la resta, hauria de ser la mitjana aritmètica entre els paràmetres dels dos virus. Això és perfecte ja que és el que ha passat !! Per exemple:

- tInc = (5+3)/2 = 4
- tImm = (8+12)/2 = 10

Annex: Codis de prova

Prova 1: ProvaIriaAfectacio

```
import java.util.ArrayList;
import java.util.List;

public class ProvaIriaAfectacio {
   public static void main(String[] args) {

       // Creem una família fictícia
       FamiliaVirus f = new FamiliaVirus("TestFam", 0.2, 0);

       // Creem un virus ARN amb les característiques de l'exemple
       VirusARN v = new VirusARN("prova", f,
               0.5, // pMal (prob de malaltia)
               3,   // tInc (temps incubació)
               3,   // tLat (temps latència)
               0.1, // pMor (prob de mort)
               4,   // tCon (temps contagi)
               0.3, // pCon (no afecta a l'exemple)
               10,  // tImm (temps immunitat)
               0.0  // pMutEC (sense mutació)
       );

       // Creo una regió amb una població que m'he inventat
       Regio r = new Regio("TestLand", 10000, 0.2);

         // Creo l'afectació amb 0 infectats inicials (els afegeixo manualment segons el
exemple que vull recrear)
       AfectacioVirusRegio afr = new AfectacioVirusRegio(v, r, 0);

       List<Integer> infectats_no_contagiosos = new ArrayList<>();
       infectats_no_contagiosos.add(200);
       infectats_no_contagiosos.add(100);
       infectats_no_contagiosos.add(90);
       infectats_no_contagiosos.add(80);

       List<Integer> contagiosos = new ArrayList<>();
       contagiosos.add(90);
       contagiosos.add(100);
       contagiosos.add(111);
       contagiosos.add(110);

       List<Integer> malalts = new ArrayList<>();
       malalts.add(42);
       malalts.add(45);
       malalts.add(55);
       malalts.add(50);

       List<Integer> morts = new ArrayList<>();
       morts.add(0);
       morts.add(0);
       morts.add(0);
       morts.add(0);
```

```
        // Forcem estat del dia D-1 segons exemple
        afr.posar_infectats_prova(infectats_no_contagiosos);
        afr.posar_malalts_prova(malalts);
        afr.posar_morts_prova(morts); // representem a, b, c com 0 per simplificar
        afr.posar_contagiosos_prova(contagiosos); // per evitar errors interns
        afr.determina_valors_prova(3040, 1515, 150);

        // Simulem 200 nous contagis (tal com diu l'exemple)
        // Imprimim resultats tal com diu l'exemple
        System.out.println("== INFECTATS NO CONTAGIOSOS INICIALS ==");
        System.out.println(afr.dona_infectats_no_contagiosos());

        System.out.println("== INFECTATS INICIALS ==");
        System.out.println(afr.dona_contagiosos());

        System.out.println("== MALALTS INICIALS ==");
        System.out.println(afr.dona_malalts());

        System.out.println("== TOTALS INICIALS ==");
        System.out.println("Total malalts: " + afr.dona_totalMalalts());
        System.out.println("Total morts: " + afr.dona_totalMorts());
        System.out.println("Total infectats: " + afr.dona_totalInfectats());

        // Ara avancem un dia
        afr.avançarUnDia();

        System.out.println("   ");
        System.out.println("AVANÇEM UN DIA :)");
        System.out.println("   ");

        // Imprimim resultats tal com diu l'exemple
        System.out.println("== INFECTATS NO CONTAGIOSOS ==");
        System.out.println(afr.dona_infectats_no_contagiosos());

        System.out.println("== INFECTATS ==");
        System.out.println(afr.dona_contagiosos());

        System.out.println("== MALALTS ==");
        System.out.println(afr.dona_malalts());

        System.out.println("== MORTS PER DIA ==");
        System.out.println(afr.dona_mortsDiaries());

        System.out.println("== TOTALS ==");
        System.out.println("Total malalts: " + afr.dona_totalMalalts());
        System.out.println("Total morts: " + afr.dona_totalMorts());
        System.out.println("Total infectats: " + afr.dona_totalInfectats());

    }
}
```

Prova 2: ProvaMutacioCoincidencia

```java
import java.util.ArrayList;
import java.util.List;

public class ProvaMutacioCoincidencia {
   public static void main(String[] args) {

       FamiliaVirus f = new FamiliaVirus("FamiliaAuladell", 0.15, 10);

       VirusARN A = new VirusARN("A", f, 0.5, 2, 1, 0.1, 3, 0.3, 5, 0.0);
       VirusARN B = new VirusARN("B", f, 0.5, 2, 1, 0.1, 3, 0.3, 5, 0.0);

       Regio r = new Regio("IriaLand", 10000, 0.2);

       r.afegirNovaAfectacio(A, 3000);  // 3000 infectats inicials d'A
       r.afegirNovaAfectacio(B, 4000);  // 4000 infectats inicials de B

       AfectacioVirusRegio afA = r.esta_present_virus_a_la_regio(A);
       AfectacioVirusRegio afB = r.esta_present_virus_a_la_regio(B);

       // INFECTATS A
       List<Integer> infectatsA = new ArrayList<>();
       infectatsA.add(3000);
       afA.posar_infectats_prova(infectatsA);

       // MALALETS, MORTS i CONTAGIOSOS A
       afA.posar_malalts_prova(new ArrayList<Integer>());
       afA.posar_morts_prova(new ArrayList<Integer>());
       afA.posar_contagiosos_prova(new ArrayList<Integer>());
       afA.determina_valors_prova(3000, 0, 0);

       // INFECTATS B
       List<Integer> infectatsB = new ArrayList<>();
       infectatsB.add(4000);
       afB.posar_infectats_prova(infectatsB);

       // MALALETS, MORTS i CONTAGIOSOS B
       afB.posar_malalts_prova(new ArrayList<Integer>());
       afB.posar_morts_prova(new ArrayList<Integer>());
       afB.posar_contagiosos_prova(new ArrayList<Integer>());
       afB.determina_valors_prova(4000, 0, 0);

       // CONTAGIOSOS A
       List<Integer> contA = new ArrayList<>();
       contA.add(3000);
       afA.posar_contagiosos_prova(contA);

       // CONTAGIOSOS B
       List<Integer> contB = new ArrayList<>();
       contB.add(4000);
       afB.posar_contagiosos_prova(contB);
```

```
        r.comprovarMutacionsPerCoincidencia();

        System.out.println("=== RESULTATS DESPRÉS DE MUTACIÓ PER COINCIDÈNCIA ===");
        for (Virus v : r.afectacionsPresentes()) {
            System.out.println("- Virus: " + v.nom());
            AfectacioVirusRegio af = r.esta_present_virus_a_la_regio(v);
            System.out.println("  Infectats: " + af.dona_infectats_no_contagiosos());
            System.out.println("  Total infectats: " + af.dona_totalInfectats());
            System.out.println("  Total infectats: " + af.nousContagios());
        }
    }
}
```

Prova3: ProvaConfinament

```
public class ProvaConfinament {
   public static void main(String[] args) {

       // ------------------------------------------------------------------------
       // ---------------------- PROVA DE CONFINAMENTS ---------------------------
       // ------------------------------------------------------------------------


        // PAS 1: Crear dues regions amb població que m'he inventat i una taxa de contacte
interna que també m'he inventat
       Regio barcelona = new Regio("Barcelona", 1000000, 0.5);
       Regio girona = new Regio("Girona", 150000, 0.4);

        // PAS 2: Les he fet veïnes i he posat per les dos una axa de contacte. El valor
dona igual, ara només vull veure
       // que quan confino passa a 0.0 i que quan desconfino passa a 0.3 un altre cop
       barcelona.afegirRegioVeina(girona, 0.3);
       girona.afegirRegioVeina(barcelona, 0.3);

       // PAS 3: Miro que s'han desat bé les taxas que he posat
       System.out.println("\n--- ABANS DE CONFINAR ---");
       System.out.println("Taxa interna BCN: " + barcelona.taxaInternaContacte());
                                System.out.println("Taxa    externa    BCN->GIR:    "    +
barcelona.taxaExternaContacte(girona));
                                System.out.println("Taxa    externa    GIR->BCN:    "    +
girona.taxaExternaContacte(barcelona));

       // PAS 4: Aplico confinament DUR a Barcelona
       barcelona.aplicarConfinamentDur(0.1);

        // PAS 5: Miro que realment les taxes han canviat (hauria de ser 0.1 la interna i
0.0 la externa)
       System.out.println("\n--- DESPRÉS DE CONFINAMENT DUR ---");
       System.out.println("Taxa interna BCN: " + barcelona.taxaInternaContacte());
                                System.out.println("Taxa    externa    BCN->GIR:    "    +
barcelona.taxaExternaContacte(girona));
                                System.out.println("Taxa    externa    GIR->BCN:    "    +
girona.taxaExternaContacte(barcelona));
       System.out.println("Barcelona confinada? " + barcelona.estaEnConfinament());

        // PAS 6: Ara el que faig és desconfinar i tornar a mirar les taxes si han tornat
que ser iguals que al princi
       barcelona.aixecarConfinament();

       System.out.println("\n--- DESPRÉS D'AIXECAR CONFINAMENT ---");
       System.out.println("Taxa interna BCN: " + barcelona.taxaInternaContacte());
                                System.out.println("Taxa    externa    BCN->GIR:    "    +
barcelona.taxaExternaContacte(girona));
                                System.out.println("Taxa    externa    GIR->BCN:    "    +
girona.taxaExternaContacte(barcelona));
       System.out.println("Barcelona confinada? " + barcelona.estaEnConfinament());
```

```
        // PAS 7: Ara en contes de fer el dur, faig el TOU només entre Girona i Barcelona
        girona.aplicarConfinamentTouAmb(barcelona);

        System.out.println("\n--- DESPRÉS DE CONFINAMENT TOU GIR->BCN ---");
                                System.out.println("Taxa    externa    BCN->GIR:    "    +
barcelona.taxaExternaContacte(girona));
                                System.out.println("Taxa    externa    GIR->BCN:    "    +
girona.taxaExternaContacte(barcelona));
        System.out.println("Girona confinada? " + girona.estaEnConfinament());

         // PAS 8: Miro que realemnt les dos tenen 0.0 i per tant, que el mètode que em diu
que estan aillades funciona
        System.out.println("\n--- COMPROVACIÓ CONFINAMENT ENTRE REGIONS ---");
                    System.out.println("Hi   ha   confinament   entre   BCN   i   GIR?   "   +
barcelona.hiHaConfinamentAmb(girona));
    }
}
```

Prova4: TestContagiAmbConfinament

```
import java.util.ArrayList;
import java.util.List;

public class TestContagiAmbConfinament {
   public static void main(String[] args) {

       // Creem una família fictícia i un virus com a l'exemple de ProvaIriaAfectacio
       FamiliaVirus f = new FamiliaVirus("TestFam", 0.2, 0);
       VirusARN virus = new VirusARN("prova", f,
               0.5,  // pMal
               3,    // tInc
               3,    // tLat
               0.1,  // pMor
               4,    // tCon
               0.3,  // pCon (no afecta)
               10,   // tImm
               0.0   // pMutEC
       );

       // Creem dues regions connectades
       Regio regA = new Regio("RegioA", 10000, 0.2);
       Regio regB = new Regio("RegioB", 10000, 0.2);
       regA.afegirRegioVeina(regB, 0.3);
       regB.afegirRegioVeina(regA, 0.3);

        // Creem l'afectació amb valors de prova i la posem manualment (igual que el de la
prova 1, he reutilitzat el codi)
       AfectacioVirusRegio afr = new AfectacioVirusRegio(virus, regA, 0);

       List<Integer> infectats_no_contagiosos = new ArrayList<>();
       infectats_no_contagiosos.add(200);
       infectats_no_contagiosos.add(100);
       infectats_no_contagiosos.add(90);
       infectats_no_contagiosos.add(80);

       List<Integer> contagiosos = new ArrayList<>();
       contagiosos.add(90);
       contagiosos.add(100);
       contagiosos.add(111);
       contagiosos.add(110);

       List<Integer> malalts = new ArrayList<>();
       malalts.add(42);
       malalts.add(45);
       malalts.add(55);
       malalts.add(50);

       List<Integer> morts = new ArrayList<>();
       morts.add(0);
       morts.add(0);
       morts.add(0);
```

```
        morts.add(0);

        afr.posar_infectats_prova(infectats_no_contagiosos);
        afr.posar_malalts_prova(malalts);
        afr.posar_morts_prova(morts);
        afr.posar_contagiosos_prova(contagiosos);
        afr.determina_valors_prova(3040, 1515, 150);

        // Afegim aquesta afectació a la regió
        regA.afegirAfectacioProva(virus, afr); // no volem sumar més infectats

        // PAS 1: Avancem un dia sense confinament
        regA.avancarDia();
        regB.avancarDia();

        System.out.println("\n--- SENSE CONFINAMENT ---");
        System.out.println("Contagiosos RegioA: " + regA.nombreContagiosos(virus));
        System.out.println("Infectats RegioB: " + regB.nombreInfectats(virus));
        System.out.println("Poblacio RegioB: " + regB.poblacio());

        // PAS 2: Apliquem confinament dur
        regA.aplicarConfinamentDur(0.1);
        System.out.println("\n[Confinament aplicat a RegioA]");

        // PAS 3: Avancem un dia més
        regA.avancarDia();
        regB.avancarDia();

        System.out.println("\n--- AMB CONFINAMENT ---");
        System.out.println("Contagiosos RegioA: " + regA.nombreContagiosos(virus));
        System.out.println("Infectats RegioB: " + regB.nombreInfectats(virus));
        System.out.println("Poblacio RegioB: " + regB.poblacio());

        // PAS 4: Aixequem el confinament
        regA.aixecarConfinament();
        System.out.println("\n[Confinament aixecat a RegioA]");

        // PAS 5: Avancem un altre dia
        regA.avancarDia();
        regB.avancarDia();

        System.out.println("\n--- DESPRÉS DE DESCONFINAR ---");
        System.out.println("Contagiosos RegioA: " + regA.nombreContagiosos(virus));
        System.out.println("Infectats RegioB: " + regB.nombreInfectats(virus));
        System.out.println("Poblacio RegioB: " + regB.poblacio());
    }
}
```

Prov 5.1: ProvaNomMutacions

```
import java.util.List;

public class ProvaNomMutacions {
   public static void main(String[] args) {


       FamiliaVirus fam = new FamiliaVirus("FamAuladell", 0.2, 20);

       // Creo un virus ARN inicial
       VirusARN virusOriginal = new VirusARN("VirusIria", fam,
               0.4, 3, 3, 0.1, 5, 0.3, 10, 0.5);

       System.out.println("Mutacions per error de còpia:");
       VirusARN actual = virusOriginal;
       for (int i = 1; i <= 5; i++) {
           VirusARN mutat = actual.mutacio();  // fem que el virus actual mutí
           System.out.println("- Mutacio " + i + ": " + mutat.nom());
       }

       System.out.println("\nMutacions per coincidència:");
       // Creem dos virus diferents de la mateixa família
       VirusARN virusA = new VirusARN("VirusMireia", fam,
               0.3, 2, 2, 0.05, 4, 0.2, 8, 0.4);
       VirusARN virusB = new VirusARN("VirusRoman", fam,
               0.6, 4, 3, 0.15, 6, 0.35, 12, 0.6);

       for (int i = 1; i <= 3; i++) {
           VirusARN virusMutat = virusA.mutacio(virusB);
           System.out.println("- Coincidència " + i + ": " + virusMutat.nom());
       }

       // Mostrem totes les mutacions creades per virusA
       System.out.println("\nMutacions registrades a virusA:");
       List<VirusARN> mutacions = virusA.obtenirMutacions();
       for (VirusARN v : mutacions) {
           System.out.println("- " + v.nom());
       }

       // Mostrem totes les mutacions creades per virusOriginal
       System.out.println("\nMutacions registrades a virusOriginal:");
       List<VirusARN> mutacionsErrorCopia = virusOriginal.obtenirMutacions();
       for (VirusARN v : mutacionsErrorCopia) {
           System.out.println("- " + v.nom());
       }
   }
}
```

Prova 5.2: ProvaCanviValorsMutacio

```
import java.util.List;

public class ProvaCanviValorsMutacio {
   public static void main(String[] args) {

       // MUTACIÓ PER ERROR DE CÒPIA

       FamiliaVirus fam = new FamiliaVirus("FamAuladell", 0, 20);  // tpcMax = 20%
          VirusARN virusBase = new VirusARN("VirusIria", fam, 0.5, 3, 3, 0.1, 4, 0.3, 10,
0.2);

       VirusARN mut1 = virusBase.mutacio();

       System.out.println("--- MUTACIÓ PER ERROR DE CÒPIA ---");
       System.out.println("Nom nou virus: " + mut1.nom());
       System.out.println("pMal: " + mut1.probDesenvoluparMalaltia());
       System.out.println("pMor: " + mut1.taxaMortalitat());
       System.out.println("tCon: " + mut1.tempsContagi());
       System.out.println("pCon: " + mut1.taxaContagi());


       // MUTACIÓ PER COINCIDÈNCIA

       VirusARN virusA = new VirusARN("VirusMireia", fam, 0.4, 3, 3, 0.2, 5, 0.2, 8, 0.2);
       VirusARN virusB = new VirusARN("VirusRoman", fam, 0.6, 5, 5, 0.1, 3, 0.4, 12, 0.3);

       VirusARN virusMutat = virusA.mutacio(virusB);

       System.out.println("\n--- MUTACIÓ PER COINCIDÈNCIA ---");
       System.out.println("Nom nou virus: " + virusMutat.nom());
       System.out.println("pMal: " + virusMutat.probDesenvoluparMalaltia());
       System.out.println("pMor: " + virusMutat.taxaMortalitat());
       System.out.println("tCon: " + virusMutat.tempsContagi());
       System.out.println("pCon: " + virusMutat.taxaContagi());
       System.out.println("tInc: " + virusMutat.tempsIncubacio());
       System.out.println("tLat: " + virusMutat.tempsLatencia());
       System.out.println("tImm: " + virusMutat.tempsImmunitat());
       System.out.println("pMutEC: " + virusMutat.probabilitatMutacioErrorCopia());
   }
}
```